

Deforming fluid domains within the finite element method

Five mesh-based tracking methods in comparison

S. Elgeti H. Sauerland

Aachen Institute
for Advanced Study in
Computational Engineering Science

Financial support from the
Deutsche Forschungsgemeinschaft (German Research Foundation)
through grant GSC 111 is gratefully acknowledged.

Deforming fluid domains within the finite element method

Five mesh-based tracking methods in comparison

S. Elgeti ^{*} H. Sauerland

Chair for Computational Analysis of Technical Systems, CCES, RWTH Aachen University, Germany

Abstract

Fluid flow applications can involve a number of coupled problems. One is the simulation of free-surface flows, which require the solution of a free-boundary problem. Within this problem, the governing equations of fluid flow are coupled with a domain deformation approach. This work reviews five of those approaches: interface tracking using a boundary-conforming mesh and, in the interface capturing context, the level-set method, the volume-of-fluid method, particle methods, as well as the phase-field method. The history of each method is presented in combination with the most recent developments in the field. Particularly, the topics of extended finite elements (XFEM) and NURBS-based methods, such as Isogeometric Analysis (IGA), are addressed. For illustration purposes, two applications have been chosen: two-phase flow involving drops or bubbles and sloshing tanks. The challenges of these applications, such as the geometrically correct representation of the free surface or the incorporation of surface tension forces, are discussed.

Keywords: free-surface flow, interface capturing, interface tracking, NURBS, XFEM

1 Introduction

In the numerical analysis of fluid flow, we often encounter free-boundary value problems: apart from the flow solution in the bulk domain, the position of (a portion of) the boundary is also unknown. This boundary can either be an external boundary or an interface between subdomains. At the boundary/interface, certain boundary conditions need to be fulfilled, which specify the position of the boundary. These conditions relate the variables of the flow (velocity, pressure, stress) across the domains under consideration of external influences, such as for example surface tension.

^{*}elgeti@cats.rwth-aachen.de

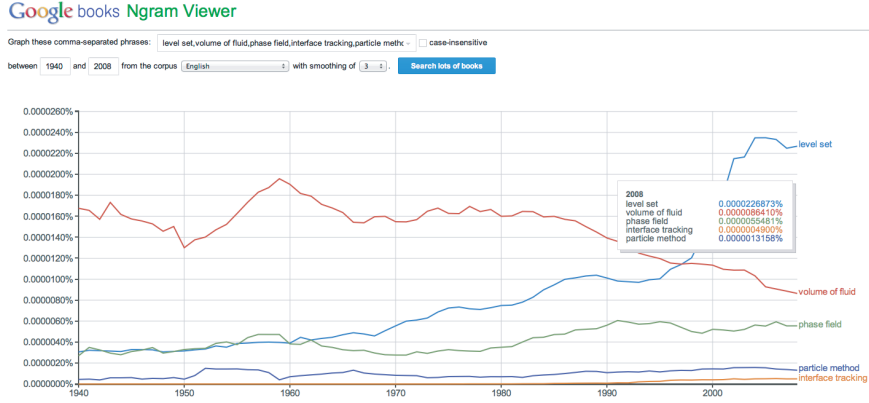


Figure 1: Google Ngram Viewer: quantitative analysis of popularity of five main interface description methods.

Numerically, in order to be able to compute the flow solution as well as the boundary/interface geometry, a measure to track the boundary starting from an initial position needs to be incorporated. Recently, Coutinho gave an overview of the most common mesh-based methods in this area – level-set, volume-of-fluid, and phase-field – from a very different point of view: a pure quantitative analysis of popularity using Google books Ngram Viewer [51]. Ngram Viewer is a tool, which analyzes all 30 million books digitalized by Google with respect to certain keywords [157]. The return value is the fraction of these books, in which the specific keyword appears. For this paper, we have added particle methods and interface tracking to the search, resulting in Figure 1. From the diagram, we get the notion that some methods already have a very long history of unremitting usage, such as the phase-field method, which – as we will later see – dates back already to 1871. The use of the level-set method rocketed within the last two decades, no doubt due to the large number of very effective recent developments. From pure numbers, it seems as if the volume-of-fluid methods may have already passed their zenith. Particle methods and interface tracking have so far not been able to compete with the more established methods on the grounds of popularity, but seem to be reserved for niche applications. Notwithstanding these crude quantifications, this paper aims to highlight the advantages and disadvantages of the above methods, illuminating the history of each method, but focusing on the most recent advancements in the individual fields.

Numerous fluid flow applications involve deformable domains: drops and bubbles, die swell, dam break, liquid storage tanks, dendritic growth, spinodal decomposition, up to and including the topic of topology optimization, which can rely on the same boundary tracking methods.

The scenario we consider here are flow solutions with either one or with two immiscible fluids. The general case is illustrated in Figure 2. Consider a d -dimensional computational domain $\Omega \subset \mathbb{R}^d$ with boundary $\Gamma = \partial\Omega$. This domain contains one, two or more immiscible fluids, which are enclosed in subdomains $\Omega_i(t)$. Position, number and shape of the individual subdomains may vary over time, i.e., both the domains themselves and the flow field are part of the solution. The main task in this context is to account for the interface Γ^{int} , which separates the distinct fluid domains and is generally in motion. Apart from its exact position, the computation of geometrical

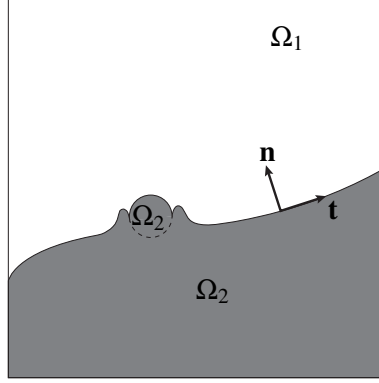


Figure 2: Illustration of the general two-phase flow scenario.

quantities of Γ^{int} , as for example its normal and tangential vectors \mathbf{n} and \mathbf{t} or its curvature κ are of interest. The corresponding computational tasks can be summarized as: (1) define the shape and location of the interface, (2) track the time advancement of the interface, and (3) set boundary conditions along the interface.

The paper is organized as follows. In Section 2, the governing equations for the fluid flow with the appropriate boundary conditions for the interface are introduced. Particular challenges that come with the numerical treatment of these equations are highlighted in Section 3. The subsequent sections concentrate on the five numerical methods considered within the scope of this paper: particle methods in Section 4, the volume-of-fluid method in Section 5, the level-set method in Section 6, the phase-field method in Section 7, and mesh-conforming interface tracking in Section 8. To illustrate the methods, two common applications, drops and sloshing tanks, are the topics of Sections 9 and 10.

2 Governing equations

In general, the governing equations for incompressible fluid flow are the instationary, incompressible Navier-Stokes equations. Consider an instationary fluid flow problem with any number (in this paper, one or two) of immiscible Newtonian phases. The computational domain at each instant in time, denoted by Ω_t , is a subset of \mathbb{R}^{nsd} , where nsd is the number of space dimensions. Then at each point in time $t \in [0, T]$, the velocity, $\mathbf{u}(\mathbf{x}, t)$, and the pressure, $p(\mathbf{x}, t)$, in each phase are governed by the following equations:

$$\rho_i \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} - \mathbf{f} \right) - \nabla \cdot \boldsymbol{\sigma}_i = \mathbf{0} \text{ on } (\Omega_t)_i \quad \forall t \in [0, T], \quad (1)$$

$$\nabla \cdot \mathbf{u} = 0 \text{ on } (\Omega_t)_i \quad \forall t \in [0, T], \quad (2)$$

for $i = 1, \dots, np$ (number of phases) and ρ_i as the density of the respective fluid. The stress tensor $\boldsymbol{\sigma}_i$ is defined as

$$\boldsymbol{\sigma}_i(\mathbf{u}, p) = -p\mathbf{I} + 2\mu_i\boldsymbol{\varepsilon}(\mathbf{u}) \quad \text{on} \quad (\Omega_t)_i, \quad (3)$$

with

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}(\nabla\mathbf{u} + (\nabla\mathbf{u})^T), \quad (4)$$

where μ_i denotes the dynamic viscosity. \mathbf{f} includes all external body forces referred to the unit mass of fluid. The computational domain Ω_t is divided into parts $(\Omega_t)_i$, each occupied by fluid i . Note that the spatial domain is time-dependent, which is indicated by subscript t . A subdomain may never be occupied by more than one fluid. The interface of two subdomains, $\partial\Omega_1 \cap \partial\Omega_2$, is denoted by Γ_t^{int} .

To allow a better grasp of the Navier-Stokes equations (1) – (2), we will illustrate the purpose of the individual terms, following in large parts the description in [46].

- **Temporal derivative $\frac{\partial\mathbf{u}}{\partial t}$** : The change in velocity with respect to time. This change over time is governed by the following influence factors:
- **Inertia term $\mathbf{u} \cdot \nabla\mathbf{u}$** : This term is a convection term arising from the conservation of momentum. The momentum of each portion of fluid needs to be conserved. Therefore, it needs to move with the fluid – it is convected with the fluid.
- **Pressure term $-\nabla p$** : This term appears when the Newtonian constitutive equation (3) is inserted into Equation (1). It includes forces resulting from pressure differences within the fluid into the formulation. Especially in the case of incompressible fluids, it is very important to consolidate this term with Equation (2).
- **Friction term $\mu_i\nabla^2\mathbf{u}$** : Again, this term arises when the Newtonian constitutive equation (3) is inserted into Equation (1). We obtain a diffusion operator equalizing the velocity of neighbouring elements. The more viscous the fluid, the stronger is the friction between neighbouring particles and thus the equalizing effect.
- **External forces \mathbf{f}** : This term includes external body forces such as gravity or vibrational excitation.

For creeping flows (i.e., Reynolds number $\ll 1$), the advective term in the Navier-Stokes equations is often neglected, giving rise to the Stokes equations. If furthermore the solution remains unaltered over time, the stationary Stokes equations can be employed:

$$-\nabla \cdot \boldsymbol{\sigma}_i = \mathbf{f} \quad \text{on} \quad \Omega_i, \quad (5)$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{on} \quad \Omega_i. \quad (6)$$

The constitutive equations for Newtonian fluids remain the same as above.

2.1 Boundary, initial, and interface conditions

In the transient case, a divergence-free velocity field for the whole computational domain is needed as an initial condition:

$$\mathbf{u}(\mathbf{x}, 0) = \hat{\mathbf{u}}^0(\mathbf{x}) \quad \text{in } \Omega \text{ at } t = 0. \quad (7)$$

In order to obtain a well-posed system, boundary conditions have to be imposed on the external boundary of Ω , denoted as Γ . Here, we distinguish between Dirichlet and Neumann boundary conditions given by:

$$\mathbf{u} = \hat{\mathbf{u}} \quad \text{on } \Gamma_u, t \in [0, T], \quad (8)$$

$$\mathbf{n} \cdot \boldsymbol{\sigma} = \hat{\mathbf{h}} \quad \text{on } \Gamma_h, t \in [0, T], \quad (9)$$

where $\hat{\mathbf{u}}$ and $\hat{\mathbf{h}}$ are prescribed velocity and stress values. Γ_u and Γ_h denote the Dirichlet and Neumann part of the boundary, forming a complementary subset of Γ , i.e., $\Gamma_u \cup \Gamma_h = \Gamma$ and $\Gamma_u \cap \Gamma_h = \emptyset$. \mathbf{n} refers to the outer normal vector on Γ_h .

In the case of two-phase flow, at the interface between the two phases, we impose interface conditions, which couple the velocity and stress between the two domains. The first interface condition implies that the velocities are continuous across the interface:

$$\mathbf{u}_1 = \mathbf{u}_2. \quad (10)$$

The second interface condition is based on the Laplace-Young equation [19]:

$$(\boldsymbol{\sigma}_2 - \boldsymbol{\sigma}_1)\mathbf{n}_1 = \gamma\kappa\mathbf{n}_1. \quad (11)$$

Here, \mathbf{n}_1 is the, with respect to Ω_1 , outward unit normal vector on Γ_t^{int} , γ the surface tension coefficient and κ the sum of the principal curvatures of Γ_t^{int} .

In addition to the interface conditions, the jump in density and viscosity between the fields leads to non-smooth behavior of the field quantities. If we examine condition (10) closer, we note that it states that the normal component of the velocity $u_n = \mathbf{u} \cdot \mathbf{n}_1$ as well as the tangential velocity component $u_t = \mathbf{u} \cdot \mathbf{t}_1$ must be continuous across the interface. Unaffected by this, it can however be shown that the normal gradient $\frac{\partial}{\partial n}$ of the tangential velocity is discontinuous [137]:

$$\frac{\partial u_{t,1}}{\partial n} - \frac{\partial u_{t,2}}{\partial n} = -(\mu_1 - \mu_2) \frac{\partial u_n}{\partial \hat{t}}, \quad (12)$$

with $\frac{\partial}{\partial \hat{t}}$ denoting the tangential derivative. In consequence, the velocity has a kink across the interface if $\mu_1 \neq \mu_2$.

Inserting the definitions for stress and strain (Equations (3) and (4)) into interface condition (11) results in:

$$[-p_1 \mathbf{I} + \mu_1 (\nabla \mathbf{u}_1 + (\nabla \mathbf{u}_1)^T) + p_2 \mathbf{I} - \mu_2 (\nabla \mathbf{u}_2 + (\nabla \mathbf{u}_2)^T)] \mathbf{n}_1 = \gamma \kappa \mathbf{n}_1. \quad (13)$$

If we restrict ourselves to only the normal component of relation (13), an expression for the pressure jump across the interface can be derived:

$$\mathbf{n}_1^T [-p_1 \mathbf{I} + \mu_1 (\nabla \mathbf{u}_1 + (\nabla \mathbf{u}_1)^T) + p_2 \mathbf{I} - \mu_2 (\nabla \mathbf{u}_2 + (\nabla \mathbf{u}_2)^T)] \cdot \mathbf{n}_1 = \gamma \kappa, \quad (14)$$

$$\Leftrightarrow -p_1 + p_2 + [2\mu_1 (\nabla \mathbf{u}_1 \mathbf{n}_1) \cdot \mathbf{n}_1 - 2\mu_2 (\nabla \mathbf{u}_2 \mathbf{n}_1) \cdot \mathbf{n}_1] = \gamma \kappa, \quad (15)$$

$$\Leftrightarrow -p_1 + p_2 + [2\mu_1 \frac{\partial u_{n,1}}{\partial n} - 2\mu_2 \frac{\partial u_{n,2}}{\partial n}] = \gamma \kappa. \quad (16)$$

From Equation (16) we can deduce that the pressure jump across the interface depends on the surface tension coefficient, the curvature of the interface, and the jump in the viscosity weighted by the normal derivative of the normal velocity component [12, 117, 124]. In particular, this has the consequence that pressure jumps across the interface are possible even when no surface tension effects are present, a possibility often disregarded in the modelling process. Usually however, this effect is indeed negligible as either the difference in viscosity or the change in the normal velocity are not high enough to actually make an impact [117].

A further influence on the pressure distribution is the volume force per unit mass of fluid \mathbf{f} . To see this, we examine the hydrostatic case, i.e., $\mathbf{u} = \mathbf{0}$, where the momentum equation (1) reduces to

$$\nabla p_i = \rho_i \mathbf{f}_i \quad \text{in } \Omega. \quad (17)$$

Focusing on the interface Γ^{int} , a jump condition for the pressure *gradient* is obtained:

$$\nabla p_1 - \nabla p_2 = \rho_1 \mathbf{f}_1 - \rho_2 \mathbf{f}_2 = (\rho_1 - \rho_2) \mathbf{f}. \quad (18)$$

The last simplification can be made since the volume force is usually constant across the entire domain. We note that we obtain a jump in the pressure gradient proportional to the jump in density, implying a kink in the pressure distribution. This behavior is extendible to $\mathbf{u} \neq \mathbf{0}$.

All effects combined (depicted in Figure 3), in the general two-phase flow scenario we have to account for a kink in the velocity as well as a jump and kink in the pressure at the interface.

2.2 Variational form

In the finite element method, we work with the variational form of the governing equations. One important measure when deriving the variational form is the integration by parts with the goal of reducing the differentiability requirements on the trial functions (i.e., the solution).

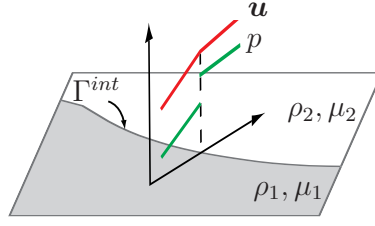


Figure 3: General situation in two-phase flow: A kink in the velocity (due to $\mu_1 \neq \mu_2$) and a jump (due to surface tension) as well as a kink (due to volume forces) in the pressure across the interface.

Integration by parts furthermore naturally includes Neumann type boundary conditions as in our case Equations (9) and (11). [113]

The variational form of Equations (1)–(2) can be expressed as follows: Find \mathbf{u} and p such that $\forall \mathbf{w}, \forall q$:

$$\begin{aligned} \sum_{i=1}^{np} \left[\int_{(\Omega_t)_i} \mathbf{w} \cdot \rho_i \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} - \mathbf{f} \right) d\mathbf{x} + \right. \\ \left. \int_{(\Omega_t)_i} \boldsymbol{\varepsilon}(\mathbf{w}) : \boldsymbol{\sigma}_i(p, \mathbf{u}) d\mathbf{x} + \int_{(\Omega_t)_i} q \nabla \cdot \mathbf{u}^h d\mathbf{x} \right] \\ = \int_{\Gamma_h} \mathbf{w} \cdot \hat{\mathbf{h}} d\mathbf{x} - \gamma \int_{\Gamma_t^{int}} \kappa \mathbf{w} \cdot \mathbf{n}_1 d\mathbf{x}. \end{aligned} \quad (19)$$

Sections 4 – 8 will introduce a variety of methods that can be employed to solve the above equation numerically.

3 The challenges in free-surface flow

In this section, we will analyze particular challenges of Equation (19): the free boundary problem it represents, the treatment of the surface tension term, as well as the solution of additional equations on the interface.

3.1 The free boundary problem

Equations (1) – (2) describe a scenario where a partial differential equation is solved for an unknown function – in this case the fluid velocity $\mathbf{u}(\mathbf{x}, t)$ and the pressure $p(\mathbf{x}, t)$ – but at the same time the exact extent of the computational domain (or portions of it) is also unknown. We are dealing with a free boundary problem. Consequently, one of the computational tasks will be to monitor the domain shape – and especially the interface/boundary – throughout the simulation. Broadly, two significant approaches can be distinguished: interface capturing and interface tracking. The main difference can be located in the viewpoint, which can be either Eulerian or Lagrangian, as we will see in the following two sections.

3.1.1 Interface Capturing

Interface capturing approaches are based on an Eulerian description and widely used in deformable domain problems. They define the interface implicitly on a fixed mesh. A characteristic scalar field ϕ is used to identify the two phases as well as the interface along the boundaries of the individual fluid domains. Depending on the different methods, this scalar field may be described for example by a discontinuous Heaviside function or a signed-distance function. In order to account for the interface motion, a standard advection equation

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0 \quad \text{in } \Omega, t \in [0, T], \quad (20)$$

is solved with \mathbf{u} as the fluid velocity. The most common representatives of this category are particle methods (Section 4), the volume-of-fluid method (Section 5), and the level-set method (Section 6). In addition, the phase-field method (Section 7) shares some features with the interface capturing methods.

A great advantage of the interface capturing approaches is that they are inherently able to account for topological changes of the interface. This allows for a much more flexible interface description than in interface tracking approaches. What needs to be considered however are the aspects of discontinuity treatment across the interface, mass conservation, and the application of boundary conditions along the interface, which remain a challenge in interface capturing.

3.1.2 Interface Tracking

In interface tracking approaches, the interface is described explicitly on a boundary/interface conforming mesh. The idea is to track the position of the mesh nodes \mathbf{x} in a Lagrangian fashion by integrating the evolution equation

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{u}(\mathbf{x}, t), \quad (21)$$

with \mathbf{u} as the fluid velocity in the domain. We can distinguish between fully Lagrangian approaches, where all mesh nodes are treated in a Lagrangian fashion, and Arbitrary Lagrangian-Eulerian (ALE) approaches, where the Lagrangian treatment is only applied to a portion of the mesh nodes, usually those along the boundary/interface (cf. Section 8).

Interface tracking approaches offer an accurate and computationally efficient approximation of the boundary/interface. Furthermore, the imposition of boundary conditions at the interface is simple with mesh nodes lying on the interface itself. However, firstly the mesh quality will usually degrade significantly in the course of large deformations and secondly topological changes require special treatment. In both cases, the last resort is remeshing with respect to the new interface. One consequence of remeshing is the need to project the field values from the old to the new mesh, a ceaseless cause for errors, and at the same time computationally costly especially in 3D. Consequently, it is desirable to keep the frequency of remeshing low (even to the level of no remeshing) [208].

3.2 Surface tension

Much of the computational accuracy and stability of two-phase flow depends on the discretization of the term responsible for the surface tension force; in the variational form Equation (19) this is:

$$\gamma \int_{\Gamma_t^{int}} \kappa \mathbf{w} \cdot \mathbf{n} \, d\mathbf{x}, \quad (22)$$

with surface tension coefficient γ , κ as the sum over all principle curvatures, \mathbf{w} denoting the test function and \mathbf{n} the normal vector. For most numerical approaches – although we will see some exceptions later – the evaluation of the curvature κ is nonviable as it contains second derivatives. As an alternative, two main strategies have evolved to avoid the direct evaluation of Equation (22): the Laplace-Beltrami technique and the Continuum Surface Force approach. These approaches are applicable irrespective of the interface discretization method. They will be detailed in the following.

3.2.1 The Laplace-Beltrami technique

The idea of the Laplace-Beltrami technique goes back to Dziuk [63] and has been closely described in [16, 87, 97]. The method replaces the above surface integral (22) requiring the curvature (and therefore the existence of second derivatives) by an integral requiring only gradient computations. In the following description, we will show two derivations of the method. The first is relevant when after discretization, an interface-conforming mesh (explicit interface representation) will be used and the second is suitable for implicit interface descriptions.

When the interface will eventually be resolved explicitly by the mesh, our starting point is the following relation regarding the curvature vector $\boldsymbol{\kappa}$ of the interface [73]:

$$\boldsymbol{\kappa} = \Delta^g \mathbf{X}, \quad (23)$$

$$\mathbf{X} : D \subset \mathbb{R}^{nsd-1} \mapsto \mathbb{R}^{nsd}, \quad (24)$$

where \mathbf{X} is an immersion, i.e., a mapping from an arbitrary parameter space to a boundary or interface. An immersion can be understood as a differentiable map between differentiable manifolds whose derivative has full rank. In the case of free-surface flows, we deal with smooth embeddings, which are injective immersions. The curvature vector $\boldsymbol{\kappa}$ always points in normal direction: It can therefore be written as $(\kappa \mathbf{n})$. We obtain:

$$\kappa \mathbf{n} = \Delta^g \mathbf{X}. \quad (25)$$

This means that we have arrived at the expression needed in term (22). Again in the case of smooth embeddings, Δ^g takes on the form of the so-called Laplace-Beltrami operator, a generalization of the Laplace operator to manifolds, whose exact definition will be given later in Equation (28). Before doing so, we define a quantity essential for any geometrical computation on our immersion \mathbf{X} : the metric \mathbf{g} given as

$$\mathbf{g} = (g_{ij}) = \mathbf{J}\mathbf{X}^T \mathbf{J}\mathbf{X} \in \mathbb{R}^{(nsd-1) \times (nsd-1)}. \quad (26)$$

The metric is intimately related to the measurement of distances and angles, and as such, also an important requisite when integrals are to be defined in such a way that they are invariant to coordinate transformations. For an arbitrary scalar function f , the coordinate transformation yields:

$$\int_{\Gamma^{int}} f(\mathbf{x}) d\mathbf{x} \stackrel{def.}{=} \int_{\Gamma_{\xi}^{int}} f(\mathbf{X}(\xi)) \sqrt{\det g(\xi)} d\xi. \quad (27)$$

Note that in the cases, where the coordinate transformation maps $\mathbb{R}^{nsd} \mapsto \mathbb{R}^{nsd}$, $\mathbf{J}\mathbf{X}$ is square. As a consequence, $\det(\mathbf{g}) = (\det(\mathbf{J}\mathbf{X}))^2$ and the definition includes the well-known transformation (change of variables) rule used routinely within the finite element method.

Within the Laplace-Beltrami technique, the general procedure in modifying the integral in (22) will now be to replace $\kappa \mathbf{n}$ by the Laplace-Beltrami operator applied to the immersion \mathbf{X} , integrate by parts and then transform the integral to reference coordinates. The definition used here for the Laplace-Beltrami operator can be found in [73]. The Laplace-Beltrami operator applied to a scalar function f is:

$$\Delta^g f = \frac{1}{\sqrt{\det g}} \partial_i (g^{ij} \sqrt{\det g} \partial_j f), \quad (28)$$

with $(g^{ij}) = \mathbf{g}^{-1}$. Inserted into the surface integral and integrated by parts we obtain:

$$\begin{aligned} \gamma \int_{\Gamma^{int}} \kappa \mathbf{w} \cdot \mathbf{n} d\mathbf{x} &\stackrel{(1)}{=} \gamma \int_{\Gamma^{int}} \mathbf{w} \cdot \Delta^g \tilde{\mathbf{X}} d\mathbf{x} \\ &\stackrel{(2)}{=} \gamma \int_{\Gamma^{int}} \mathbf{w} \cdot \partial_i (g^{ij} \sqrt{\det \tilde{g}} \partial_j \tilde{\mathbf{X}}) d\tilde{\xi} \\ &\stackrel{(3)}{=} -\gamma \int_{\Gamma_{\xi}^{int}} \left(\frac{\partial}{\partial \tilde{\xi}_i} \mathbf{w}^k \right) g^{ij} \left(\frac{\partial}{\partial \tilde{\xi}_j} \tilde{\mathbf{X}}_k \right) \sqrt{\det \tilde{g}} d\tilde{\xi} \\ &\stackrel{(4)}{=} \sum_{\text{interface elements}} -\gamma \int_I \left(\frac{\partial}{\partial \xi_i} \mathbf{w}^k \right) g^{ij} \left(\frac{\partial}{\partial \xi_j} (\mathbf{X}^e)_k \right) \sqrt{\det g} d\xi. \end{aligned} \quad (29)$$

The steps in Equation (22) are detailed in the following:

- (1) The curvature vector is replaced using the Laplace-Beltrami technique.
- (2) We insert definition (28) for the Laplace-Beltrami operator. The integral is transformed to a parametrization of the interface in ξ -coordinates as given in Figure 4. This parametrization is still smooth.
- (3) Under the assumption that \mathbf{w} is sufficiently smooth, integration by parts can be performed.

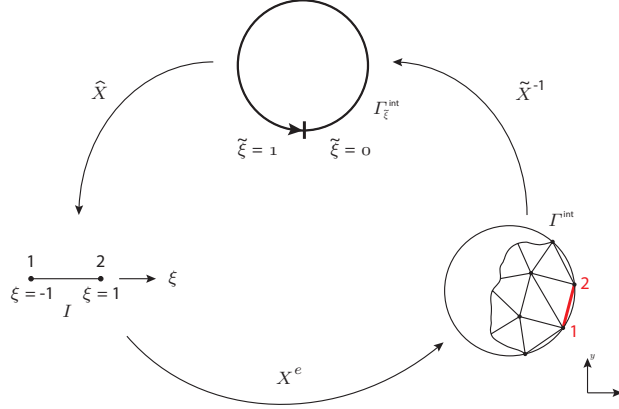


Figure 4: The interface Γ^{int} , which is given in \mathbf{x} -coordinates, is reparametrized with the parameter $\tilde{\xi} \in [0, 1]$. This is a smooth embedding and the Laplace-Beltrami technique can be applied. Partial integration can be performed. The integrals are then as usual computed on the reference element in ξ coordinates. The 1D reference element is mapped to a portion of the boundary of a 2D mesh (here, we see an excerpt).

- (4) The interface is discretized using a boundary conforming finite element mesh. The integral is computed element-wise through a reparametrization into element reference coordinates ξ .

For the integration by parts, we assume that the surface is closed and therefore, the boundary term vanishes. This is an assumption regularly made [87, 97].

In order to give an example of the actual implementation, we will demonstrate how to evaluate the immersion term in Equation (29) in a standard finite element setting. \mathbf{X}^e is defined as the mapping from reference coordinates ξ to physical coordinates \mathbf{x} :

$$\mathbf{X}^e : \xi \mapsto \mathbf{x} \quad \mathbf{X}^e(\xi) = \sum_{k=1}^{nen} N_k(\xi) \mathbf{x}_k^e. \quad (30)$$

Here, N_k denotes the finite-element shape function for node k and nen indicates the number of nodes per element. Note that we refer to the boundary element as indicated in Figure 4.

The derivative of \mathbf{X} with respect to the (here only one) reference coordinate ξ evaluated at a specific point ξ_a is then:

$$\left. \frac{\partial \mathbf{X}}{\partial \xi} \right|_{\xi_a} = \sum_{k=1}^{nen} \left. \frac{\partial N_k}{\partial \xi} \right|_{\xi_a} \mathbf{x}_k. \quad (31)$$

This concludes the explicit case.

In the implicit case, the starting point is to express the sum of principle curvatures of the interface Γ^{int} by a relation similar to Equation (25) as [97, 87]:

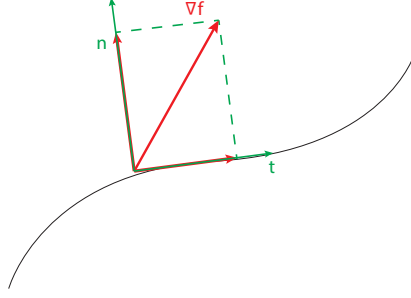


Figure 5: The tangential gradient of a function f can be evaluated by computing the full gradient and subsequently subtracting the normal component of the gradient.

$$\kappa \mathbf{n} = \Delta_{\Gamma} id_{\Gamma^{int}}. \quad (32)$$

In Equation (32), $id_{\Gamma^{int}}$ signifies the identity mapping on Γ^{int} defined as $id_{\Gamma^{int}}(\mathbf{x}) : \Gamma^{int} \mapsto \Gamma^{int} := \mathbf{x} = (x_1, x_2, \dots, x_{nsd})^T$. Note that this definition implies that $\nabla(id_{\Gamma^{int}})$ are exactly the basis vectors of \mathbb{R}^{nsd} . Furthermore, Δ_{Γ} denotes again the Laplace-Beltrami operator. However in this variant, a different definition is employed. This is also the reason, why for now a different symbol has been chosen. Apart from the definition in Equation (28), it can also be defined using the tangential derivative and the tangential divergence [97]:

$$\Delta_{\Gamma} = \nabla_{\Gamma} \cdot \nabla_{\Gamma}, \quad (33)$$

where the tangential derivative ∇_{Γ} of an arbitrary function f is defined as:

$$\nabla_{\Gamma} f := \nabla f - (\mathbf{n} \cdot \nabla f) \mathbf{n} = (\mathbf{I} - \mathbf{n} \mathbf{n}^T) \nabla f, \quad (34)$$

and the tangential divergence of \mathbf{f} is defined as :

$$\nabla_{\Gamma} \cdot \mathbf{f} := \nabla \cdot \mathbf{f} - \mathbf{n}^T (\nabla \mathbf{f}) \mathbf{n}. \quad (35)$$

Using these relations to replace $\kappa \mathbf{n}$, we can again take advantage of the idea that integration by parts within the weak form can be used in order to reduce the order of derivatives. Again, the integral in equation (22) is restricted to the interface and therefore the integration by parts is not straightforward. In the case of a closed surface Γ^{int} , where the boundary term vanishes, this leads to [97]:

$$\gamma \int_{\Gamma^{int}} \kappa \mathbf{w} \cdot \mathbf{n} \, d\mathbf{x} = -\gamma \int_{\Gamma^{int}} \nabla_{\Gamma} \mathbf{w} : \nabla_{\Gamma} id_{\Gamma^{int}} \, d\mathbf{x} = -\gamma \int_{\Gamma^{int}} tr((\mathbf{I} - \mathbf{n} \mathbf{n}^T) \nabla_{\Gamma} \mathbf{w}) \, d\mathbf{x}. \quad (36)$$

The assumption of a closed surface is valid in many applications where surface tension is important, such as for example drops and bubbles. Consult [97] for derivations for open boundaries and modifications for moving interfaces. Note that as compared to the derivation in the explicit case, the extraction of the tangential gradient using $(\mathbf{I} - \mathbf{n}\mathbf{n}^T)$ is necessary. This stands contrary to the definition in Equation (29), where all gradients already point in tangential direction.

The obvious question to pose at this point is how the two presented expressions for the Laplace-Beltrami operator, Δ^g in Equation (28) and Δ_Γ in Equation (33), can be related. In the context of this method, the crucial difference between the explicit and the implicit approach is that in the explicit case, the test function \mathbf{w} can be expressed directly in terms of the reference coordinates $\boldsymbol{\xi}$; even along the interface. In the implicit case, however, the test function on the interface $\mathbf{w}(\boldsymbol{\xi})$ needs to be expressed in terms of the test function in the bulk domain, which we will denote as $\tilde{\mathbf{w}}(\mathbf{x})$:

$$\mathbf{w}(\boldsymbol{\xi}) = \tilde{\mathbf{w}}(\mathbf{X}(\boldsymbol{\xi})), \quad (37)$$

$$\tilde{\mathbf{w}} : \mathbf{X}(D) \subset \mathbb{R}^{nsd} \mapsto \mathbb{R}^{nsd}. \quad (38)$$

Over the course of the next equations, we will demonstrate that despite this difference between the approaches, Equations (29) and (36) are to be considered as equivalent. If we insert definition (37) into Equation (29), which has so far only been used for the explicit case, we obtain

$$\begin{aligned} & -\gamma \int_I \left(\frac{\partial}{\partial \xi^i} \mathbf{w}^k \right) g^{ij} \left(\frac{\partial}{\partial \xi^j} (\mathbf{X}^e)_k \right) \sqrt{\det g} \, d\boldsymbol{\xi} \\ &= -\gamma \int_I \left(\frac{\partial}{\partial \xi^i} (\mathbf{X}^e)^l \frac{\partial}{\partial x^l} \tilde{\mathbf{w}}^k \right) g^{ij} \left(\frac{\partial}{\partial \xi^j} (\mathbf{X}^e)_k \right) \sqrt{\det g} \, d\boldsymbol{\xi} \\ &= -\gamma \int_I \left(\frac{\partial}{\partial x^l} \tilde{\mathbf{w}}^k \right) \underbrace{g^{ij} \left(\frac{\partial}{\partial \xi^i} (\mathbf{X}^e)^l \frac{\partial}{\partial \xi^j} (\mathbf{X}^e)_k \right)}_{(\mathbf{P}(\boldsymbol{\xi}))_k^l} \sqrt{\det g} \, d\boldsymbol{\xi} \\ &= -\gamma \int_I \text{tr} \left(\nabla \tilde{\mathbf{w}}|_{\mathbf{X}^e(\boldsymbol{\xi})} \mathbf{P}(\boldsymbol{\xi}) \right) \sqrt{\det g} \, d\boldsymbol{\xi}. \end{aligned} \quad (39)$$

Based on both the definition of the normal \mathbf{n} and $(g^{ij}) = (g_{ij})^{-1}$

$$\mathbf{P}\mathbf{n} = 0, \quad (40)$$

and

$$\mathbf{P} \frac{\partial}{\partial \xi^i} \mathbf{X}^e = \frac{\partial}{\partial \xi^i} \mathbf{X}^e, \quad (41)$$

hold. In other words, \mathbf{P} is a projection onto the tangent space. Since in the 2D case $\{\mathbf{n}, \frac{\partial}{\partial \xi^1} \mathbf{X}^e\}$ and in 3D $\{\mathbf{n}, \frac{\partial}{\partial \xi^1} \mathbf{X}^e, \frac{\partial}{\partial \xi^2} \mathbf{X}^e\}$ form a basis, Equations (40) – (41) define \mathbf{P} unambiguously. In particular, this means that

$$\mathbf{P} = \mathbf{I} - \mathbf{nn}^T. \quad (42)$$

With Equation (33) and $\mathbf{P}^2 = \mathbf{P}$ we obtain

$$\begin{aligned} & -\gamma \int_I \left(\frac{\partial}{\partial \xi^i} \mathbf{w}^k \right) g^{ij} \left(\frac{\partial}{\partial \xi^j} (\mathbf{X}^e)^k \right) \sqrt{\det g} d\xi = \\ & -\gamma \int_I \text{tr}(\mathbf{P} \nabla_\Gamma \tilde{\mathbf{w}}) \sqrt{\det g} d\xi = -\gamma \int_{\Gamma^{int}} \text{tr}(\mathbf{P} \nabla_\Gamma \tilde{\mathbf{w}}) d\Gamma. \end{aligned} \quad (43)$$

The latter is equivalent to Equation (36).

Note that in all cases of both explicit and implicit boundary descriptions, the assumption that Equations (22) and either (29) or (36) are completely equivalent only holds if Γ^{int} is sufficiently smooth. This will in general not hold for the discretization of Γ^{int} [97].

3.2.2 Continuum Surface Force approach

Another alternative to Equation (22) is the Continuum Surface Force approach (CSF) as introduced in [32]. This is of particular interest when modelling topologically complex interfaces. The basic idea is to replace the interface coupling condition (11) by a localized volume force term in the momentum equation (1). The force term has the following form [96]:

$$\mathbf{f}_\Gamma = \gamma \kappa \delta_\Gamma \mathbf{n}. \quad (44)$$

Here, δ_Γ refers to a smoothed Dirac δ -function needed to select the surface force to the proximity of the interface, i.e., it selects a narrow band around the interface. \mathbf{f}_Γ is designed in such a way that it exactly replicates the surface tension force per interfacial area that the surface integral would generate. Away from the interfacial region, the surface force is 0. Throughout the transition region, all fluid quantities vary continuously.

3.3 Solving additional equations on the interface

In some applications, it may be necessary to solve additional partial differential equations on the interface/boundary: an example for the topic of solving partial differential equations on hypersurfaces (manifolds). One prominent example for such a case is the solution of the transport equation (advection-diffusion equation) as an additional equation to the governing equations of fluid mechanics [83, 119, 140], materials science [58, 76], and biology [17, 134, 145]. The main topic is the consideration of insoluble surfactants (surface active agents), which adhere to the phase interface and influence the fluid flow in an either restraining or stimulating way. One type of surfactant known to all of us in our everyday lives is soap. It is important to note, that the surfactant is located exclusively at the interface/boundary and never in the bulk domain (cf. Figure 6). Surfactants appear either as pollutants, which have accidentally entered the flow, or are purposefully added for example to decrease the surface tension and thereby increase the ability of a liquid to wet solid surfaces [56].

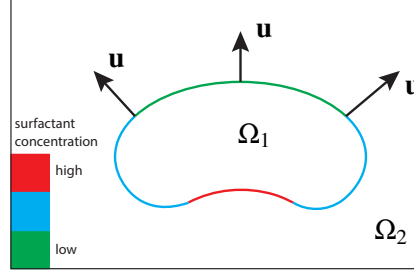


Figure 6: Depending on the flow field and the diffusion coefficient, the surfactant concentration on the interface of, e.g., a drop, can be determined in every time step.

The two main driving forces for the transport are advection (movement of the interface) and diffusion (molecular diffusion along the interface) [97]. Even though the governing equations for the scalar transport processes are often known and well understood, solving the respective partial differential equation on arbitrary and even moving manifolds is not straightforward. The effects brought about by the embedding, as already mentioned in Section 3.2.1 have to be accounted for. The solution approaches differ significantly depending on whether an explicit or implicit mesh description has been chosen (cf. Section 3.1). In the explicit approach, an interface mesh can be extracted from the bulk mesh thus easily defining the domain on which the transport equation is to be solved. In order to solve the proper equation, the differential operators of the partial differential equation have to be generalized in order to account for the arbitrarily shaped hyper-surface [64, 83]. Note that during the integration on the reference domain, it is mandatory to include the metric introduced in Equation (27). Implicit approaches [2, 29, 65, 169, 221] typically extend the scalar field away from the interface to the whole domain in order for all differential operators to be defined properly. One possible approach to the extension can be found in [36, 41]. This obviously increases the dimension of the problem, but the transport equation can then be solved in the whole fixed computational domain. [97] provides a good overview of different Eulerian and Lagrangian approaches.

The solution of the transport equation requires a coupling of the flow solution with the transport. In addition, in many applications we would like to consider the effects that the surfactant has on the interface; i.e., a coupling of the transport solution to the flow equations. This coupling enters through the surface tension coefficient γ , which can be considered to be a function of the surfactant concentration G . This leads to tangential forces along the interface, the so-called Marangoni convection. [83, 97]

The standard advection-diffusion equation would be given as:

$$\frac{\partial G}{\partial t} + \nabla \cdot (\mathbf{a}G) - \nabla \cdot (\mathbf{d}\nabla G) = 0, \quad (45)$$

where \mathbf{a} denotes the (possibly time dependent) advection velocity and \mathbf{d} the diffusion coefficient matrix. In the case, where the above equation needs to be solved on a deformable hypersurface, some modifications need to be taken into account [83, 97]:

1. In the surfactant context, we will usually assume that \mathbf{d} is constant and isotropic. Therefore $\nabla \cdot (\mathbf{d} \nabla G) = \mathbf{d} \Delta G$.
2. Since the equation is to be solved on an interface embedded into a larger domain, the differential operators in Equation (45) need to be replaced by the corresponding operators restricted to the interface: the gradient ∇ is replaced by the tangential gradient ∇_Γ (cf. Equation (34)), the divergence $\nabla \cdot$ by the tangential divergence $\nabla_\Gamma \cdot$ (cf. Equation (35)) and the Laplace operator Δ by the Laplace-Beltrami operator Δ_Γ (cf. Equations (28) and (33)).
3. The advection velocity – and in the case of surfactant transport the advection velocity will be the fluid velocity \mathbf{u} – needs to be restricted to its tangential component $\mathbf{u}_\Gamma = (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\mathbf{u}$.
4. The term $\nabla_\Gamma \cdot (\mathbf{u}G)$ can be rewritten as

$$\nabla_\Gamma \cdot (\mathbf{u}G) = G \nabla_\Gamma \cdot \mathbf{u} + \mathbf{u}_\Gamma \cdot \nabla_\Gamma G. \quad (46)$$

In the cases of deformable interfaces, the first term ensures conservation of mass even under local changes in the free-surface area. Note that it is important to use \mathbf{u} instead of \mathbf{u}_Γ . Otherwise, a change in surface area would not be possible. If the interface is fixed, this term is 0 as $\mathbf{u} \cdot \mathbf{n} = 0$.

This leads to the final equation for the surfactant concentration G :

$$\frac{\partial G}{\partial t} + (\mathbf{u}_\Gamma \cdot \nabla_\Gamma)G - \Delta_\Gamma G + G \nabla_\Gamma \cdot \mathbf{u} = 0. \quad (47)$$

The above equation may be rewritten in many ways [2, 64, 83], making it more accessible to the different numerical schemes.

4 Particle Methods

Particle methods utilize mass-less particles distributed in an Eulerian mesh to capture the fluid flow and in particular the interface position. The most prominent particle method is the Marker-And-Cell (MAC) method developed by Harlow and Welch [102]. It is one of the first examples of an interface capturing method; the numerous examples presented in [102] attest to the method's extraordinary flexibility.

The original approach was developed for a single phase – i.e., $i = 1$ in Equations (1) – (2) – thus modelling a fluid surrounded by air. In later versions, the method could be extended to multi-phase flow. In all cases, the particle methods solve the “one-fluid” version of the Navier-Stokes equations, where the surface tension force is included directly and the fluid properties, e.g., ρ and μ , vary according to an indicator function.

The method covers with a computational mesh the maximal possible fluid domain and subsequently distributes marker particles throughout the entire domain currently covered with fluid (cf. Figure 7). Any cell not containing marker particles is identified as an empty cell. Cells with at least one marker particle and at least one common boundary with an empty cell are the interface

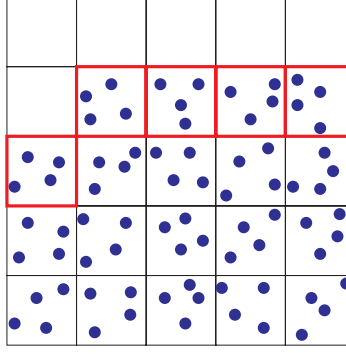


Figure 7: In the Marker-and-Cell method by Harlow and Welch [102], the following convention is used for the identification of the fluid domain and the empty domain: Any cell not containing marker particles (blue dots) is identified as an empty cell. Cells with at least one marker particle and at least one common boundary with an empty cell are the interface cells (marked in red). Cells accommodating at least one marker particle and furthermore surrounded only by other cells containing marker particles are marked as fluid cells.

cells. As a last category, cells accommodating at least one marker particle and furthermore surrounded only by other cells containing marker particles are marked as fluid cells.

Throughout the simulation, the marker particles are advanced with the local fluid velocity interpolated from the Eulerian grid.

The MAC method is strongly connected to finite-difference discretizations. In principle, however, it can also be combined with other methods, such as the finite element method in [90].

In the MAC method, the velocity and pressure solutions are usually obtained sequentially. First, a preliminary velocity field that we will denote with \mathbf{u}^{mom} is computed including all effects of the momentum equation (Equation (1)) except for the pressure: Of the terms listed in Section 2, the inertia term, the friction term and the external forces contribute. The relevant equation reads (adapted from [1] and [46]):

$$\frac{\partial^h \mathbf{u}^{mom}}{\partial^h t} = -(\mathbf{u}^n \cdot \nabla^h) \mathbf{u}^n + \mathbf{f}^n + \nu \Delta^h \mathbf{u}^n. \quad (48)$$

Here, the superscript h on the operators indicate that they are discretized in an appropriate fashion, e.g., with finite-difference schemes. The superscript n denotes the value of the corresponding field at the previous time step. Note that Equation (48) can either be solved in one step or subdivided into several steps, computing the contributions of the individual terms one by one as done in [46].

The computed \mathbf{u}^{mom} will most likely not be divergence free and therefore, Equation (2) is not fulfilled yet. However, the contribution of the pressure term to the velocity field has not been considered so far. This contribution is still available to adjust the velocity field by setting the pressure values appropriately. \mathbf{u}^{mom} and its correction factor $-\nabla p$ are inserted into Equation (2):

$$\nabla^h \cdot \left(\underbrace{\frac{\mathbf{u}^{mom}}{\Delta t}}_{known} - \underbrace{\frac{1}{\rho} \nabla^h p}_{unknown} \right) = 0. \quad (49)$$

Again, the superscript h refers to a discretized operator. Equation (49) is a Poisson-type equation for the pressure. By solving this equation, the pressure values p^{n+1} and with those, \mathbf{u}^{n+1} can be obtained. This procedure is comparable to projection methods – or Chorin-Temam method – utilized in the context of the finite element method ([74] and references therein).

Notable is the development of SMAC (simplified-MAC) [6] only five years after the original MAC implementation, which reduces the complexity of the original approach particularly when it comes to incorporating different types of boundary conditions. Its secret lies in the computation of the pressure field not based on a preliminary velocity field, but rather a potential function.

4.1 Properties

With regard to the typical applications in fluid flow, MAC has the advantage that the tracer particles are indistinguishable from one another: The joining and separation of fluid parts can be effortlessly simulated. Nevertheless, it suffers from drawbacks especially regarding the description of free surfaces. Stress effects at the free surface, such as surface tension and contact angles are difficult to include. This is particularly true since, if used with finite difference, MAC is entwined with a discretization on staggered grids, i.e., storing velocity on cell faces and the pressure at the cell center, which has been found to provide higher stability [46]. Boundary conditions at the free-surface can then for example only be set as pressures at the cell center of all cells identified as interface cells [66], i.e., not at the actual boundary, or they may require complex surface fitting techniques such as those introduced in [55]. In addition, it is difficult to compute normals and curvature from the particles [138] and to control particle drifting [224]. In general, the method can suffer from stability problems, which may be difficult to detect, with apparently reasonable particle distributions resulting from gross approximation errors. Some modification were proposed to stabilize the MAC method in [34]. McKee [154] identifies two further consequential restrictions: MAC could not be applied to arbitrary domain shapes and was, for a long time, restricted to two space dimensions. The latter was due to lacking efficiency in the solution methods for the corrected pressure, preventing the use of a large number of particles. In addition, care had to be exerted on the choice of time-stepping (no marker is allowed to cross more than one cell per time-step) [154].

4.2 Recent advances

Recalling the drawbacks listed in the last section, two main points have recently been addressed in the area of MAC: the restriction to simple shapes in 2D and the topic of applying boundary conditions at walls and on the free surface.

A development in the area of the former are GENSMAC [209] and GENSMAC3D [210]. This extension provides the applicability of MAC to domains of arbitrary shape and dimension as opposed to the straight lines required in the original implementation. This approach has

been applied to viscoelastic extrudate swell in [162, 211, 212], i.e., an application where the components of the stress tensor appear as additional unknowns in addition to the only primary variables of velocity and pressure. [190] proposes an extension to an arbitrary number of phases. In [46] a reduction of the mesh size and thus computational effort is proposed. The procedure is illustrated in Figure 8.

Boundary conditions on interfaces remained a challenge to MAC for decades. The only way to overcome this challenge was so far the combination with other methods, such as Lagrangian meshes (cf. Section 8), level-set methods (cf. Section 6) or volume-of-fluid (cf. Section 5) [154]. Santos [190] succeeded by including an additional Lagrangian mesh to track the free surface. [10] works with an ordered list of connected surface markers, which are advected along the streamlines of the flow field using a Runge-Kutta integration. Since the flow field will most likely disturb the homogeneity of the marker distribution (concentration of streamlines or vortical flow), they are subsequently added, deleted and redistributed. [224] proposes a hybrid approach in such a sense, that the standard MAC method with its usual Eulerian grid is used for the fluid domain. In addition, special marker particles identify the interface, which is subsequently reconstructed using a Lagrangian mesh. Surface tension was also considered in [193], where the authors compared an implementation using a Lagrangian mesh with an implementation based on a particle level-set method. In [138], a level-set approach is combined with Lagrangian marker particles as a means of level-set reinitialization. [135, 136] present a method where the interface is tracked using mesh-less particles. The particles are explicitly connected to certain points of an underlying Eulerian reference grid. Connectivity among the particles is not required however. In each step, the interface is resampled based on a local reconstruction of the interface, on the one hand redistributing the particles and on the other hand updating the connecting points in the reference grid. The approach taken in [218] is based on the point-set method first introduced in [213]. It uses an indicator field (a function which is either 0 or 1 depending on the corresponding fluid domain), which is constructed from massless marker particles. Subsequently, the normals and curvatures can be computed from the indicator function, which has been smoothed using the reproducing kernel particle method (RKPM). A completely different approach has been adopted in [132], where all gradient terms in the Navier-Stokes equations are computed based on the interaction between all particles within a certain kernel. This method has been extended with more efficient solvers in [123].

The MAC-method served as inspiration for the volume-of-fluid method described in the next section.

5 The Volume of Fluid Method

The volume-of-fluid (VOF) method was developed in 1981 by Hirt [105]. At the time, he worked in an environment where there were two options available to represent surfaces: either marker particles (cf. Section 4) or height functions combined with line segments. Height functions combined with line segments – Hirt gives [108] and [164] as references – were the first attempts to represent the actual interface. If the height function is used, the height of the interface as compared to a given reference line is stored for each discrete point in the domain. Problems occur for multiple-valued height functions as they would be seen for drops and when the interface slope

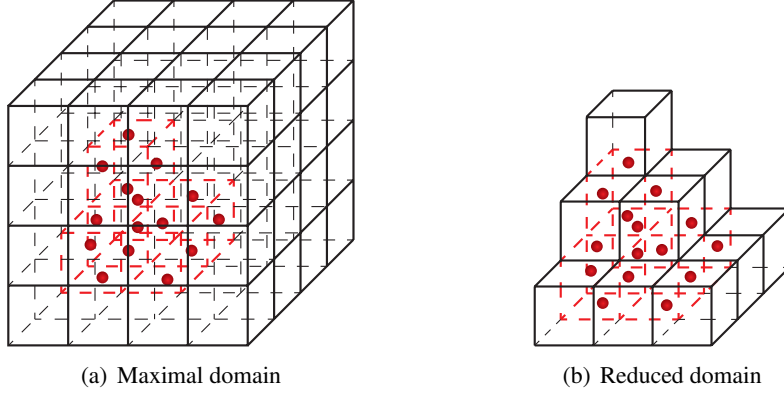


Figure 8: (a) In the original MAC approach, the maximal possible fluid domain (black cells) was meshed, even if only a minor portion of cells (red cells) contain fluid. (b) In more recent approaches, the mesh consists only of the cells containing fluid plus an additional layer of ghost cells. This is sufficient as there is a time-step requirement, which guarantees that fluid particles cannot move through more than one cell per time step. [46]

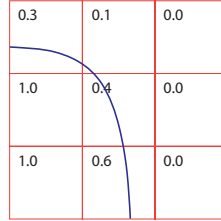


Figure 9: Excerpt of a domain containing nine cells: The interface is indicated in blue. The area left of the interface is filled with fluid. In the VOF method, for each cell, the fraction of fluid contained in the cell is stored. Cells that are completely filled with fluid have the value 1. Cells without any fluid have the value 0.

exceeds the grid resolution. The height function approach can be generalized to ordered chains of line segments. Although the two given issues with the height function are resolved now, this approach has its drawbacks when it comes to curve intersections or an extension to 3D.

As described in the previous section, the MAC method does not define the interface, but rather the fluid regions, thus leading to no logical problems during interface intersection as well as a straightforward extension to 3D – apart from the excessive storage requirements, which were a concern to Hirt. Building on this scenario, he proposed the VOF method. This successor of the MAC concept replaces the discrete marker particles with a global field defining the location of the fluid(s). This global field is called the volume-fraction field F , sharply dividing fluid ($F = 1$) and empty ($F = 0$) areas. For each cell, F takes on one specific value $0 \leq F \leq 1$, indicating the amount of fluid contained in each cell (cf. Figure 9). One way of defining F is the use of a characteristic scalar field ϕ . In the case of the VOF method, $\phi = \mathcal{H}(\mathbf{x}, t)$, with \mathcal{H} as the

heaviside function. The volume fraction for each cell Ω can then be defined as:

$$F(\Omega, t) = \frac{1}{|\Omega|} \int_{\Omega} \mathcal{H}(\mathbf{x}, t) d\Omega. \quad (50)$$

With only one value per cell, the storage requirements for this method are significantly reduced as compared to the MAC approach. A very important feature of the VOF method is its inherent mass conservation [33].

The advancement of the interface in time is governed by the following advection-type equation:

$$\frac{\partial F}{\partial t} + \nabla \cdot F = 0. \quad (51)$$

Due to its discontinuous character, the volume fraction function cannot be advected directly. Instead, it is connected with the mass conversation equation of the Navier-Stokes equations, thus including the volume-fraction field into the flow solution. The resulting equation can be reduced to the computation of fluxes across the element faces. In its most basic version, the VOF method has three major difficulties:

1. the computation of shape derivatives (as for example needed for curvature evaluation) is obstructed by the use of a non-smooth interface representation,
2. as the volume fraction function is advected, the interface becomes more and more diffuse,
3. imposition of boundary conditions along the interface is impossible [105, 7].

One possible solution to all three problems is the reconstruction of a sharp interface in each time step. Options for this reconstruction will be discussed in the following section. Nevertheless, the VOF method is still utilized today in its original version, as for example in [44].

5.1 Recent Methods for Interface Reconstruction

Many advances in the VOF method are based on the aspect of reconstruction of the interface line. Over the past decades, numerous approaches have evolved. This large number of approaches already hints towards one major problem: As the interface shape is inferred from the volume data it is never unique. The reconstructed shapes are in general either piecewise linear or piecewise constant. [188]

In the original implementation of VOF, a method that would later become known as SLIC (Simple Linear Interface Calculation) [165] was proposed, where the interface is reconstructed using line segments, which can be either parallel or perpendicular to the major flow axis. The direction of the line segment is defined through the coordinate direction in which a larger change in the fluid volume occurs (cf. Figure 10).

A second reconstruction option is PLIC (Piecewise Linear Interface Calculation) – [7, 33] and references therein. In this method, the interface is represented by a discontinuous chain of segments each obeying the definition [7]:

$$\mathbf{x}\mathbf{n} = \alpha. \quad (52)$$

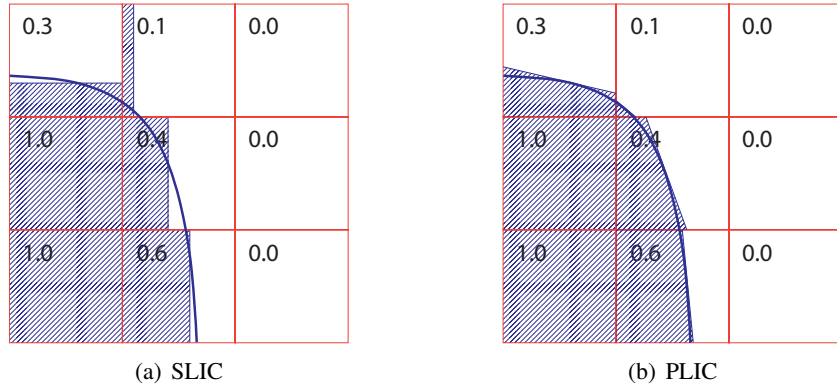


Figure 10: The two main interface reconstruction approaches for the VOF method – SLIC and PLIC – are compared. (a) In the SLIC approach, the interface is reconstructed using line segments, which can be either parallel or perpendicular to the major flow axis. (b) In the PLIC approach, the interface is represented by a discontinuous chain of segments.

The parameter α can be determined using the condition of mass conservation. A central question for this scheme is the way, in which the normal is evaluated. The obvious choice, evaluating the normal the gradient of the volume fractions $\mathbf{n} = \nabla F / |\nabla F|$ approximated through finite differences as proposed in [223] does not perform well [156], which brought up a variety of alternative choices [179]. [156] presents a generalization of PLIC to adaptive moving grids, [118] to arbitrary unstructured grids. Note that, although based on piecewise linear functions, the PLIC approach cannot in general represent linear interfaces. [188] proposes an appropriate extension. The definition of the orientation of the line segments can be enhanced by spline interpolation of the interface in every cell [142]. A special challenge is the treatment of more than two fluids, which connect at a triple point within a cell. In equilibrium, this point is connected to the contact angles of the particular fluids. In the instationary case, [33] derives a relation to the normal vectors obtained within a PLIC setting.

All piecewise linear reconstructions, i.e., both SLIC and PLIC, suffer from the unphysical creation of droplets disconnected from the actual surface by pure construction of the numerical method [147, 127].

A similar approach, using a local height function, has been introduced in [127] and was for example utilized in [147] and references therein: The method operates on 3×3 blocks of cells. After the orientation of the surface has been determined based on the values of the volume fraction field of surrounding cells, the surface height is determined row- or column-wise. Then, instead of updating the volume fraction values, the height function is updated. A subsequent adjustment step of the corresponding volume fraction values is required.

A very fruitful alternative is the coupling of VOF with the level-set approach as it will be described in Section 6 – Coupled Level-Set Volume-of-Fluid (CLSVOF). Commencing with [205], the idea has been propagated to combine the level-set approach – with its smooth interface – and the VOF approach – with its inherent mass conservation. Both the level-set function and

the volume fraction are advected simultaneously, yet individually. The volume fraction interface description is then the basis for a correction of the zero level-set [7]. Such approaches, illustrated in Figure 11, have for example been explored in [218, 9]. A similar method is the reconstructed distance function (RDF) in [54], except that here the distance function is deduced from the piecewise linear reconstruction of the volume fractions.

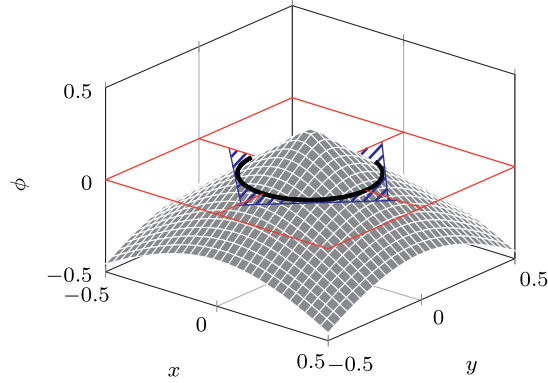


Figure 11: Coupled Level-Set Volume-of-Fluid: Both the level-set function and the volume fraction are advected simultaneously, yet individually. The volume fraction interface description is then the basis for a correction of the zero level-set (cf. Section 6).

For some applications, simplified approaches that do not rely on a full interface construction can also lead to satisfactory results. For example, [7] names WLIC (Weighted Linear Interface Calculation) [222, 150] and THINC/SW (Tangent of Hyperbola Interface Capturing with Slope Weighting) [220]. WLIC is a simplification of PLIC. In a first step it treats each coordinate of the normal vector separately, thus computing fluxes along coordinate directions. In a second step these fluxes are averaged to account for slanted normal vectors.

5.2 Curvature Computation

The general definition of the curvature κ in the VOF context is:

$$\kappa = \nabla \cdot \left(\frac{\nabla F}{|\nabla F|} \right). \quad (53)$$

However, being based on the heaviside function, the volume fraction function F is not smooth enough to be differentiated directly. Options can be found in either reconstruction of the interface (as described in the previous section), or some type of smoothing.

An alternative formulation is:

$$\kappa = \nabla \cdot \mathbf{n}. \quad (54)$$

Firstly, we will discuss methods for curvature computation that are based on interface reconstruction. In [127], Kleefsman et al. suggest to compute the curvature from the height function h using the relation:

$$\kappa = \frac{\partial}{\partial y} \left(\frac{\partial h / \partial y}{\sqrt{1 + (\partial h / \partial y)^2}} \right). \quad (55)$$

The derivatives of the height function needed here are approximated using finite differences. Improved versions are among others offered by [143], who introduced a correction scheme based on local error estimation, and [139], whose scheme involves a collection of possible height functions and corresponding curvatures, which are then sampled using quality statistics. In the context of PLIC, [155] uses a local approximation of the curvature with polynomials. The polynomial coefficients are obtained from a least-squares fit. Raessi addresses the problem from a different perspective. In [184], unit normals are advected, which are the principal constituent for both the reconstruction of the interface and the curvature calculation.

Secondly, we will address the topic of smoothing. In [54], three different approaches to curvature approximation are compared: (1) In convolved VOF (developed in [219]), F is replaced by a smoothed version \tilde{F} , which has been mollified by a smoothing kernel K . K could for example be defined as $K(r) = (1 - r^2)^4$ for $r < 1$. The danger here is that the smoothing can be so strong that the curvature effects are diminished. (2) An improvement with respect to error, but not with respect to convergence behavior, was found using RDF. (3) Superior to both method was however a height function approach by Sussman [204]. Another commonly used approach to circumventing interface reconstruction is to employ the continuum surface force (CSF) model as described in Section 3.2.2. Very recently, Baltussen has suggested the tensile force method [15]. This is based on a substantially different surface tension model: the tensile force model introduced in [214]. In this model, the surface tension force is determined as a sum of tensile forces, i.e., the force neighbouring cells exert on each interface element. This method does not require any curvature information. In [60] a method the authors name CELESTE is presented. It revolves around a second-order Taylor expansion of F from any given cell to its neighbouring cell. It is constructed from an overdetermined system of equations, utilizing a least-squares fit to determine both the normals and the curvature.

An interface capturing method which, contrary to VOF, provides a smooth interface is the level-set method, which will be subject of the next section.

6 The Level-Set Method

The level-set method is an interface capturing strategy. Level set methods [173, 35] mitigate difficulties associated with a discontinuous volume-fraction function by using instead a smooth level set function ϕ , separating areas filled with fluid A ($\phi < 0$) from those filled with fluid B ($\phi > 0$). The interface Γ^{int} is then given by:

$$\Gamma^{int} = \{\mathbf{x} \in \Omega | \phi(\mathbf{x}, t) = 0\}. \quad (56)$$

In most cases, the level-set function ϕ is defined as a signed distance function [172]:

$$\phi(\mathbf{x}, t) = \pm \min_{\mathbf{x}^* \in \Gamma_t^{int}} \|\mathbf{x} - \mathbf{x}^*\|, \quad \forall \mathbf{x} \in \Omega. \quad (57)$$

At each point \mathbf{x} and t , the level-set function ϕ stores the shortest distance from the point to the current interface. Figure 12 illustrates the concept by example of a circle.

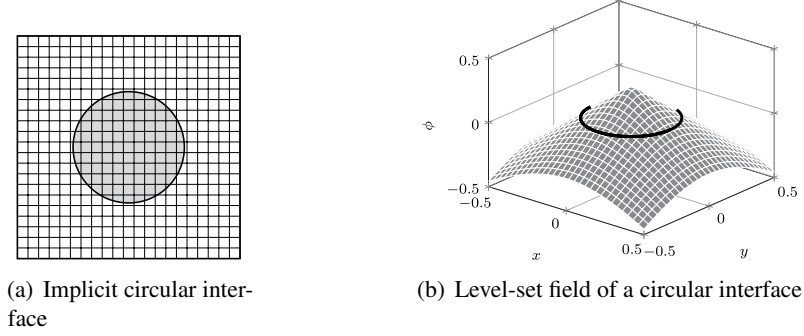


Figure 12: A 2D domain containing a circular interface: In (a), an implicit definition of the interface is shown. Note that the elements of the mesh do not conform with the interface. The implicit description is realized through a signed distance level-set function outlined in (b).

The method requires the definition of an initial level-set field from the original interface position:

$$\phi(\mathbf{x}, 0) = \phi^0(\mathbf{x}) \quad \in \Omega. \quad (58)$$

Furthermore, at inflow boundaries, the level-set field has to be defined throughout the entire time domain, thus assuring that the level-set function is well-defined at boundary points, where information is transported into the spatial domain. During the progress of the simulation, the initial ϕ is then transported with the fluid velocity $\mathbf{u}(\mathbf{x}, t)$ using the following transport equation:

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0 \quad \text{in } \Omega, \quad t \in [0, T]. \quad (59)$$

The value of the level-set function indicates, which density and viscosity values ρ_i and μ_i are associated with which element, or, along the interface, only part of an element. To illustrate the latter case, Figure 13 shows a sample element, which is cut by the interface. The interface is approximated (e.g., linearly) through the intersection points with the element edges. Depending on the side of the interface where an integration point is located, the material property associated with it is chosen:

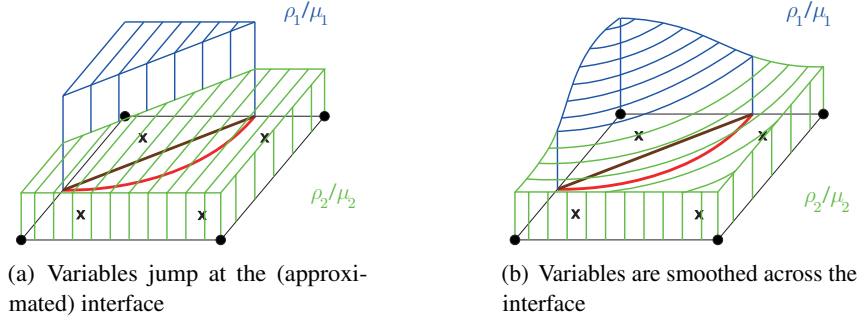


Figure 13: A quadrilateral element with four nodes is intersected by the interface (red): The interface is approximated linearly (brown). (a) During integration, all Gauss points \mathbf{x} are associated with either the properties of fluid 1 (ρ_1, μ_1) or of fluid 2 (ρ_2, μ_2). (b) As an alternative, smooth transition of the properties can be generated throughout the element. Note that in both cases, an integration point might lie in the area between the interface as indicated by the level-set function and the linearized interface. One consequence of the approximation is then that this integration point will be associated with the wrong material domain.

$$\rho, \mu(\phi) = \begin{cases} \rho_1, \mu_1, & \phi(\mathbf{x}, t) < 0; \\ \rho_2, \mu_2, & \phi(\mathbf{x}, t) > 0. \end{cases} \quad (60)$$

If standard discretization methods are used, this procedure may lead to stability problems. It is therefore suggested to create a smooth transition of the material properties through the element. This can for example be achieved by including a smoothed Heaviside function into the formulation [206]. The price to pay is, however, a smeared out interface where it was once infinitesimally thin.

Already in its original set-up, the level-set method attains the main advantage of the interface-capturing methods: the topological flexibility. Furthermore, as opposed to other interface capturing methods, the evaluation of geometrical quantities is straightforward. As ϕ is a smooth function, it can be used directly to compute the interface curvature, normals, etc. What is left is to cope with the challenges that this approach contains: treatment of discontinuities across the interface, ensuring mass conservation, applying boundary conditions on the interface, and evaluating equations on the interface.

Note that the volume-of-fluid method of the previous section can be interpreted using the same scheme, except that $\phi = \text{Heaviside}$.

6.1 Mass conservation

6.1.1 Reinitialization

Since the velocity field entering the transport equation (59) is usually non-uniform, it will over time degrade the signed-distance property of the level-set function. Even though the interface

is still represented correctly, this has severe influence on the quality of the computation of geometrical properties from the level-set function. Consequently, frequent reinitialization, i.e., restoring of the signed-distance character of ϕ , is recommended.

Different approaches for reinitialization can be found in literature [115]. In direct reinitialization approaches, the closest distance to the interface is determined individually for each node. The approach is based on identifying the intersections between the interface and element edges. Out of the collection of these intersection points, the Euclidean distance to the nearest point is computed for each node. This value is stored as the new level-set value for the node, keeping the sign intact. Unfortunately, this approach can quickly lead to efficiency problems. The efficiency can be improved by restricting the reinitialization to a narrow band around the interface [43, 177, 215] (cf. Figure 14). As an alternative, fast search tree methods can be employed to organize the distance evaluations [149]. Fast marching methods compute the distances sequentially from low to high, each computation building on the previously computed distances. [191] features timings for the different approaches.

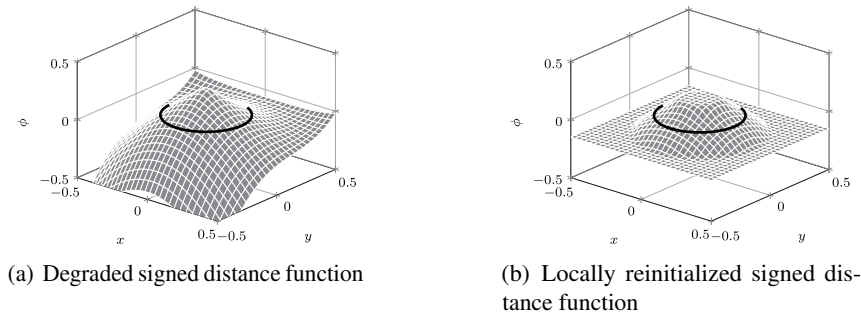


Figure 14: To handle the degraded (as compared to Figure 12) level-set function (a), a reinitialization restricted to a band around the interface is performed (b).

6.1.2 Mass correction

One very undesirable side-effect of the reinitialization procedure is the mass loss: The reinitialization does not preserve the location of the interface. This effect is enhanced through the choice of temporal and spatial discretization of the transport equation (59) [187]. Investigated solution approaches are manifold. Some approaches aim at the minimization of the displacement during reinitialization [103, 163, 206]. In [52] an iterative mass correction is described, which provides global mass conservation. As mentioned in the Sections 4 and 5, another possibility lies in the combination with particle or volume-of-fluid methods.

6.2 Obtaining sharp interfaces

With the implicit definition through the level-set function, the discontinuity conditions along the interface will usually occur inside of elements, where, as described before, a smearing of the bulk properties is recommended. One possibility that allows to account for a sharp interface within the

element is the extended finite element method (XFEM). XFEM is then often connected to the other improvement option: adaptive mesh refinement.

6.2.1 XFEM - the extended finite element method

In the finite element method, the unknown function is usually interpolated with polynomial basis functions that are continuous within the elements and C^0 continuous across element faces. This approach is very much suitable for smooth unknown functions, but leads to problems if jumps occur. The XFEM circumvents this deficiency by local and well-targeted enrichment of the basis function space. The origin of XFEM lies in fracture mechanics [24, 161]. Overviews of the method can be found in [26, 79]. In XFEM, a sample function g is approximated on a fixed mesh as:

$$g^h(\mathbf{x}, t) = \underbrace{\sum_{i \in I} N_i(\mathbf{x}) g_i}_{\text{standard FE}} + \underbrace{\sum_{i \in I^*} N_i^*(\mathbf{x}) \psi(\mathbf{x}, t) a_i}_{\text{enrichment}}. \quad (61)$$

$N_i(\mathbf{x})$ and $N_i^*(\mathbf{x})$ are the standard finite element shape functions for node i with the nodal unknown g_i . The set I is the set of all nodes in the domain, whereas I^* contains all enrichment nodes. They are enriched using the enrichment functions $\psi(\mathbf{x}, t)$, which are connected to additional XFEM unknowns a_i . Refer to Figure 15 for a sample mesh.

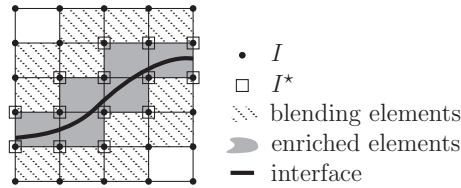


Figure 15: Sample domain with an interface. The three different element types of the XFEM: regular, enriched, and blending elements, are indicated.

An enrichment function that is typically chosen for strong discontinuities is the sign-enrichment:

$$\psi_{\text{sign}}(\mathbf{x}, t) = \text{sign}(\phi(\mathbf{x}, t)) = \begin{cases} -1, & \phi(\mathbf{x}, t) < 0, \\ 0, & \phi(\mathbf{x}, t) = 0, \\ 1, & \phi(\mathbf{x}, t) > 0. \end{cases} \quad (62)$$

In the fluid flow problems considered in this paper, the scenario is as follows: Across the interface, a jump in density and viscosity, a kink in the velocity, as well as both a jump and kink in the pressure need to be considered (cf. Figure 3).

Chessa and Belytschko [38, 39] use a kink enrichment for only the velocity approximation. Depending on whether surface tension is considered, a kink or jump enrichment of the pressure is added [78, 159, 185, 225]. Kölke [131] enriches both spaces with a jump. Even so it seems

like a natural enrichment to account for the kink in the velocity field by appropriately enriching the basis, [49] states that this does not lead to an improved overall quality of the solution. [192] even reports stability problems if this is included. As a consequence, several authors apply an enrichment for only the pressure [37, 48, 95].

It is to be noted that the enrichment also entails special quadrature procedures. In order to be able to invoke standard Gauss rules, the interface cells are subdivided into triangles as indicated in Figure 16 [24, 79, 161].

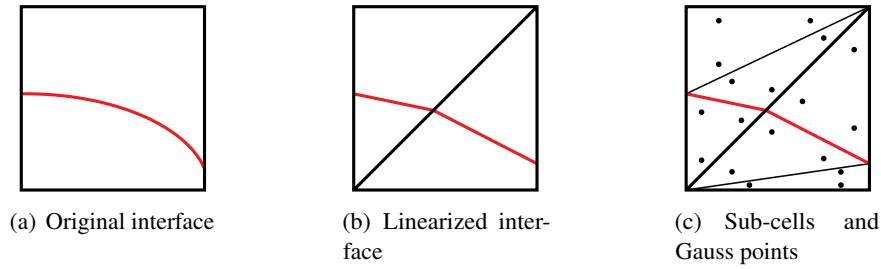


Figure 16: Linearization of a curved interface in a quadrilateral reference element by decomposing the quadrilateral into linear triangular sub-cells, where finally the integration points are placed.

6.2.2 Adaptive mesh refinement

An alternative or an addition to XFEM can be found in adaptive mesh refinement (AMR). AMR can be used to increase the solution quality in the vicinity of the interface: the most interesting part of the computational domain. While global mesh refinement usually leads to prohibitively large mesh sizes, AMR provides an alternative at reasonable computational cost [14, 180]. In [37, 80], an adaptive refinement approach tailored to XFEM is introduced. Here, the level-set functions serves as an indicator for the refinement area. The interface elements are successively subdivided up to a desired level. Figure 17 depicts the general refinement procedure for a simple setting. Here, two levels of refinement are applied. In a first step, the elements of the initial mesh, Figure 17(a), which are in contact with the interface, are subdivided once (cf. Figure 17(b)). Those elements from the first refinement, which are cut by the interface are refined a second time, see Figure 17(c). Subsequently, further elements are subdivided step by step such that neighbouring elements differ by at most one level of refinement (cf. Figure 17(d)). This assures a smooth variation of the element sizes. The level-set indicator field is defined on a fine background mesh with a mesh density corresponding to the highest refinement level (cf. Figure 17(e)). Thereby, no interpolation of the level-set values at nodes created during refinement is required and the accuracy of the interface does not degrade.

6.3 Computing geometrical quantities from the interface

Since the level-set function ϕ is smooth, it can be used to compute geometric entities. As an alternative, the (linear) approximation of the level-set function within the elements could be

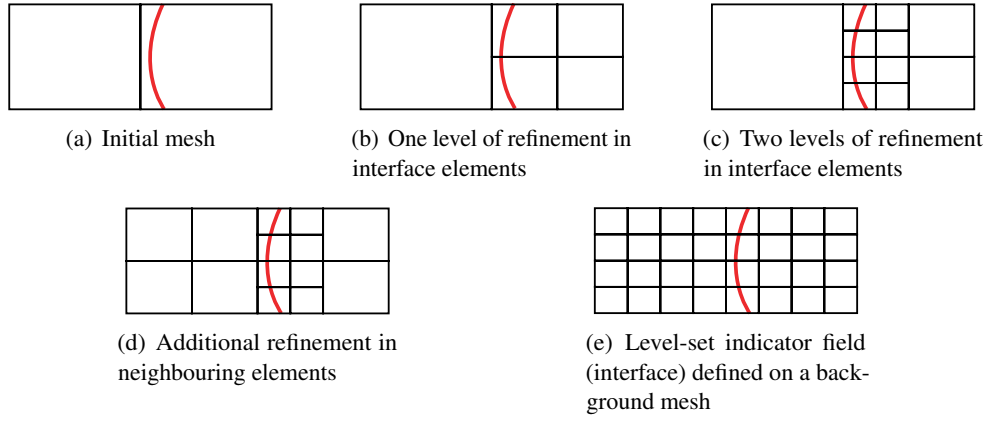


Figure 17: Adaptive mesh refinement with respect to the interface (bold red line).

employed. In terms of convergence order, [97, 96] suggest the second option.

6.3.1 Normal vectors

The normal vector is defined as:

$$\mathbf{n} = \frac{\nabla \phi(\mathbf{x}, t)}{\|\nabla \phi(\mathbf{x}, t)\|} \Big|_{\Gamma^{int}}. \quad (63)$$

Due to the signed distance property of ϕ , the denominator in (63) could essentially be dropped. However, since during the time-stepping, the level-set function is only an approximation of a signed distance function (cf. Section 6.1), it is advisable to keep the normalization.

6.3.2 Curvature

The curvature can be computed from the level-set function as:

$$\kappa = \nabla \cdot \frac{\nabla \phi(\mathbf{x}, t)}{\|\nabla \phi(\mathbf{x}, t)\|} \Big|_{\Gamma^{int}}. \quad (64)$$

This formulation is restricted to approaches, where the level-set function is not approximated linearly. An example is given in [170], where a quadratic approximation in combination with a filtering technique is employed. However, even when theoretically possible, Marchandise points out in [149] that it is in general not advisable to use the thereby computed curvature to evaluate the surface tension. One popular option is to resort to the Laplace-Beltrami technique already described in Section 3.2.1.

Particularly in combination with the XFEM, level-set provides very sharp interface descriptions. A contrasting approach is the phase-field approach described in the next section, which deliberately introduces diffuse interfaces.

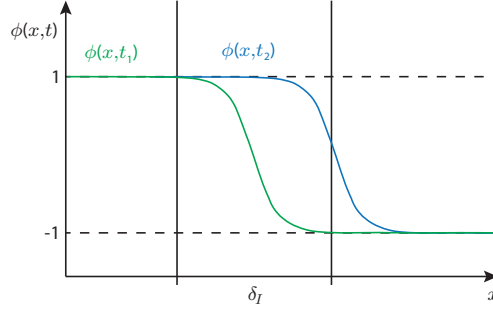


Figure 18: Illustration of the phase-field variable: $\phi(x, t)$ takes on one value (e.g., 1 in this case) in one phase and another value (in this case -1) in the second phase. The transition is rapid, but not sharp. We speak of a diffuse interface with interface width δ_I .

7 Phase-Field Method

One of the oldest numerical methods in multi-phase problems is the phase-field method. According to [70], as early as 1873, the work of Gibbs on thermodynamics already served as a foundation [89].

The main difference compared to the previously introduced interface-capturing methods is that the phase-field method works with diffuse interfaces — i.e., the transition layer between the phases has a finite size. There is no tracking mechanism for the interface, but the phase state is included implicitly in the governing equations. The interface is associated with a smooth, but highly localized variation of the so-called phase-field variable ϕ . In two-phase problems, ϕ is a scalar value; if more phases are present, it can become vector-valued. ϕ is assigned a distinct value, e.g., -1 , in phase A and another distinct value, e.g., 1 , in the other phase B. The interface could then be assumed to be located at $\phi = 0$ [30]; however, the phase-field equations never require the knowledge of the exact interface location. ϕ will vary with space and time. Typically, we observe small variations in the bulk phases and rapid variations close to the interface [70].

Comparing Figure 18 and Figure 12 gives the notion, that phase-field resembles level-set in many ways. Indeed, they have been applied for identical applications, however, there are some fundamental differences that should be considered in the choice of application. Major differences are: (a) The phase-field approach avoids setting continuum boundary conditions at the interface, which would usually be required to match the individual solutions of the bulk phases. Note also that one of these boundary conditions usually defines the advection velocity of the interface. What is particularly difficult about these boundary conditions is that they have to be set at a boundary whose shape and position is also part of the solution. Instead, these boundary conditions are substituted by the inclusion of the phase-field variable into the system. Consequently, the phase-field method is particularly useful in cases, where the continuum equations for a sharp interface do not (yet) exist. Sharp-interface descriptions can be interpreted as a phase-field approach with infinitely small interface width. However, it is not considered to be very useful to use phase-field in such a sense [70]. (b) In the phase-field method, the interface is not advected specifically as would be the case for level-set; moreover, the interface location is never computed explicitly [70, 30].

Phase-field can be seen as a computational method, in which the presence of a non-sharp interface dampens instabilities, or it can be seen as a physically motivated approach, where the governing equations do not impose sharp interface conditions [70].

The derivation of the governing equations for a system described by a phase-field approach is strongly tied to the notion of thermodynamic equilibrium. In engineering applications, we usually deal with open or closed systems; and hardly with isolated systems. In open or closed systems, the thermodynamic equilibrium is not defined through the state of maximal entropy, as it would be for isolated systems, but through the extrema of some other thermodynamic state variable. In principle, we distinguish between four variations of the equilibrium by thermodynamic forces: mechanical, thermal, material, and chemical. [146]

- *Mechanical*: Two systems under different pressures are inclined to reach mechanical equilibrium: mechanical work is performed. In such a case, the pressure is interpreted as a thermodynamic potential.
- *Thermal*: Two systems with different temperature are inclined to reach thermal equilibrium: heat flow is induced. In such a case, the temperature is interpreted as a thermodynamic potential.
- *Material*: A system containing either different material states of one material, e.g., solid and liquid phase, or two chemically inert materials, will tend to the material equilibrium. In such a case, the chemical potential is interpreted as the thermodynamic potential.
- *Chemical*: A system containing substances, which are chemically reactive, will develop a specific equilibrium between educts and products. Again, the chemical potential is interpreted as the thermodynamic potential.

The state of equilibrium is characterized through the minimum of the thermodynamic potential P . The condition for equilibrium is defined as $dP = 0$.

So far, we have stated that in order to describe a thermodynamic equilibrium of any kind, the definition of a suitable thermodynamic potential is essential. In realistic problems, several potentials interact. Therefore, the need for finding the extremum of one specific potential is now replaced by minimizing the free energy functional of the system. The free energy is the portion of the energy that is available to perform thermodynamic work. After performing such work, it is irreversibly lost [70]. Depending on the applications, different definitions of a free-energy functional can be used: the Gibbs free energy formulation is for example applicable to systems where the temperature and the pressure remain constant, whereas the Helmholtz free energy formulation can be employed for situations where the temperature and the volume remain constant. In general we can write [70]:

$$P(X_1, X_2, \dots, X_n) = \int_V P(X_1, X_2, \dots, X_n) dV, \quad (65)$$

where P refers to the appropriate thermodynamic potential with P signifying the respective density function. X_1, X_2, \dots, X_n refer to the governing variables of the system, i.e., the independent state variable. Depending on the system, these could for example be temperature, pressure, the

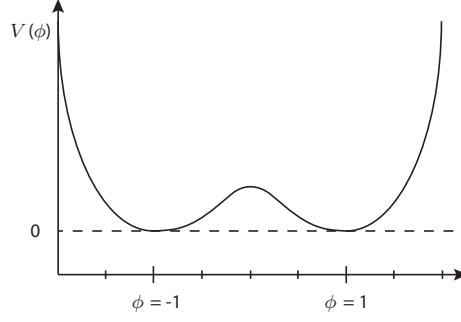


Figure 19: Double-well potential: A potential depending on the phase-field variable ϕ is defined, which fulfills the basic criteria of symmetry and two minima.

concentration of the phases, etc.. The free-energy functional together with the equations of motion leads to a full description of the system in the sense that the value of all state variables can be determined from the free-energy via extremal principles if we assume thermodynamic equilibrium. In detail, this means that through differentiation with respect to the individual variables, equations for all those variables can be obtained (cf. [70] and references [171, 57, 99, 181] therein). In the phase-field method, an additional, in a sense artificial, dependence on the phase-field variable is generated. Equation (66) now reads [70]:

$$P(X_1, X_2, \dots, X_n, \phi) = \int_V P(X_1, X_2, \dots, X_n, \phi) dV, \quad (66)$$

where ϕ is the phase-field variable.

In order to include ϕ into the formulation, a contribution of ϕ to the density function P has to be defined: the potential $V(\phi)$. $V(\phi)$ shall be constructed in such a way that it features two minima – one for each of the two phases. This ensures that exactly these two phases (e.g., liquid and solid) are the most stable states. Furthermore, the potential needs to be symmetric. If we expand $V(\phi)$ in a Taylor series, and break off after the first two terms, we will find the definition [70]:

$$V(\phi) = \frac{V_0}{4}(\phi^2 - 1)^2. \quad (67)$$

Here, it is assumed that the two minima have an energy level of 0. In order to be able to also represent inhomogeneous states, the potential also has to depend on the gradient of ϕ . The lowest order terms invariant under both rotation and translation are either $\nabla^2 \phi$ or $(\nabla \phi)^2$. The contribution of ϕ to the energy can then be expressed as:

$$P = \dots + \frac{\varepsilon}{2}(\nabla \phi)^2 + V(\phi), \quad (68)$$

with ε as a constant. This formulation is termed the Cahn-Hilliard potential.

Again using variational principles, evolution equations can be derived for the phase field.

In non-equilibrium state, two important equations that can be developed from the Gibbs free-energy functional are the Cahn-Hilliard and the Allen-Cahn equation. The latter is a reaction-

diffusion equation and in general not of importance for flow problems. The Cahn-Hilliard equation reads [30]

$$\frac{\partial C}{\partial t} = \nabla \cdot \left(M_C \nabla \left(\frac{\partial f}{\partial C} - \varepsilon_C^2 \Delta C \right) \right). \quad (69)$$

In the above equation, the unknown quantity is the concentration field C of one of the two quantities present in the simulation (e.g., fluid A and fluid B or fluid and solid). C can be interpreted as equivalent to ϕ . $f(C, \dots)$ denotes a free energy. It depends on the concentration, but also on other independent state variables such as possibly the temperature. M_C is the mobility related to the solute diffusion coefficient and ε_C is a constant related to the length scale of the diffuse interface. Note that the Cahn-Hilliard equation contains a fourth derivative of the concentration field; a challenge to many discretization methods.

The most important areas of application of the phase-field method are solidification processes, especially including phase segregation in binary alloys, and dendritic growth [67, 93, 198, 199, 203]. It has also been applied to other scenarios, such as spinodal decomposition – the unmixing of a mixture of liquids or solids – [13], surfactant transport [5, 71], or even topology optimization [59].

In [140], isogeometric analysis (cf. Section 8.4.2) is utilized to analyze liquid-vapor phase transition phenomena modelled through the Navier-Stokes-Korteweg equations, which can be categorized as a phase-field model; in [91] the same approach is used for the solution of the Cahn-Hilliard equations in the context of spinodal decomposition. For both applications, the isogeometric analysis approach profits from the arbitrarily high continuity of the finite-element basis functions across element interfaces. For the first time, this allows for a standard use of the finite element method not augmented by mixed methods or discontinuous or continuous/discontinuous Galerkin methods [91].

The discussion of the phase-field method concludes the topic of interface-capturing approaches. In the next section, interface tracking will be introduced.

8 The Interface-Tracking Approach

In interface tracking, the boundary or interface is explicitly resolved by the computational mesh. The necessity of accurately following the deforming boundary of the domain requires some degree of Lagrangian description to be present in the formulation, at least in the vicinity of the boundary. Since the use of a Lagrangian viewpoint is unavoidable, some researchers considered fully Lagrangian formulation for the entire domain [104, 168, 183, 106, 168]. The simplicity of such solutions is offset by several difficulties. The flow field in the interior of the domain may exhibit a large amount of circulation or shear, which are often not directly related to the motion of the boundary. The fully Lagrangian approach then results in unnecessary mesh distortion and invalid meshes in such interior regions, even for moderate or null displacements of the boundary. However, internal circulation and shear are handled without difficulty by an Eulerian description. This provided a strong motivation for the development of methods capable of combining the Lagrangian and Eulerian approaches in the same domain. The ALE (Arbitrary Lagrangian-Eulerian) formulation was initially stated in the finite difference context by Hirt [107], and later

adopted also in the finite element community—see [61, 25, 112, 110] and references contained therein. The basic idea of ALE is to work with a deforming mesh, as in the Lagrangian method, but to decouple the mesh deformation from the displacement of the fluid particles – or in other words, the mesh velocity is no longer equal to the fluid velocity, but can be chosen arbitrarily. The usual choice in ALE is to track boundary and interface nodes in a Lagrangian manner (or a manner that comes very close), while all other mesh nodes are adjusted solely with the intention of preserving mesh quality as best as possible. The ALE approach requires that the mesh velocity enters the momentum equation (1) explicitly, thus modifying the original flow equations, specifically the convection term. Furthermore, this equation now needs to be written over a reference domain that may or may not coincide with either material or spatial domains. This added complexity is one of the main drawbacks of ALE. Nevertheless, it is used actively in the area of fluid flow computation, e.g., in [42, 82, 85, 217]. The ALE philosophy found another expression in the family of space-time finite element methods, which are described further below.

8.1 Deforming spatial domain space-time finite element methods

The classical choice of space and time discretization within the finite element community is to use finite elements (FE) in the spatial domain, but finite differences (FD) in the time direction. One alternative are the so called space-time finite element methods, which use finite element approximations for both space and time. Leaving the spatial discretization unaltered as compared to the classical approach, an additional coordinate direction – the time t – is introduced for both the computational domain as a whole, but also for each finite element. As a consequence, all volume integrals in Equation (19) are now integrals not only over the spatial domain Ω , but over the space-time domain Q of dimension $nsd + 1$. The shape functions N_i , which interpolate the unknown functions, are equipped with an additional dependency on the time: $N_i(\mathbf{x}, t)$.

Although it is in principle possible to employ the space-time discretization with an interpolation, which is continuous-in-time, the modern space-time approach usually relies on discontinuous in time discretization in the spirit of the *Discontinuous Galerkin* method [61, 111, 101]. In order to avoid unmanageable number of degrees of freedom that must be solved for at any given time, the space-time approach is typically applied to subsets of the temporal domain called space-time slabs (cf. Figure 20), which are roughly comparable to time steps in the FE/FD approach. Due to the discontinuous in time interpolation, each node stores two values of every unknown per point in time. In order to enforce a relation between these two values, the continuity of the solution in temporal direction is imposed weakly by adding a jump term to the variational form (19).

The space-time method offers the unique opportunity to write the variational form directly over a deforming domain. This concept was pioneered by Jamet and Bonnerot [121, 31, 77, 148, 120]. It was later extended concurrently by Tezduyar et al. [207] in the form of the Deformable-Spatial-Domain/Stabilized Space-Time (DSD/SST) method, also applied in [23, 160], and by Hansbo [100] with a particular mesh moving scheme conforming to the characteristic streamline diffusion ideology. Note that in the deforming domain space-time approach, there is no need to explicitly include the mesh velocity into the variational form, as it would be mandatory for ALE.

In most space-time implementations to date, the meshes for the space-time slabs are simply extruded in the temporal direction from a spatial mesh, resulting in reference element domains that are always Cartesian products of spatial and temporal reference domains. Such an approach is

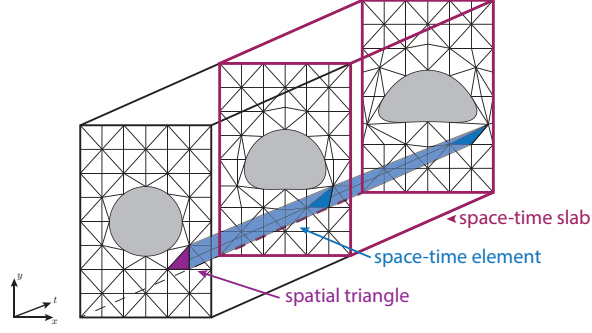


Figure 20: Illustration of two successive space-time slabs $\hat{\Omega}$. From the lower time level to the upper time level, the mesh can deform. In blue, one space-time element is exemplified. It is triangular in space and forms a prism in space-time.

best described as semi-structured (unstructured in space, structured in time) and does not leave the option of increasing temporal refinement in portions of the domain. In such a case, the space-time slab exactly corresponds to a time step of a semi-discrete procedure. In fact, many stencils of the FE/FD methods may be re-derived by using the semi-structured space-time approach with appropriate weighting and interpolation functions.

Note that the applicability of the space-time approach is of course not limited to the deforming domains in an interface-tracking context. Extensions of the XFEM concept to space-time [40, 175, 151], the monolithic space-time fluid-structure-interaction solvers [109], or shape optimization [68, 176] are just a few examples.

8.2 The mesh deformation

In interface tracking, the mesh is continuously adapted to the flow. Procedures for mesh deformation are described in the following, categorized into boundary deformation and displacement of the inner nodes.

8.2.1 Displacement of the boundaries

Section 3.1.2 describes the standard choice for the motion of the domain interface/boundary in the interface tracking context: a full Lagrangian deformation as specified by Equation (21). This translates to a deformation, where the mesh velocity \mathbf{v} is chosen exactly equal to the fluid velocity \mathbf{u} :

$$\mathbf{v}(\mathbf{x}) = \mathbf{u}(\mathbf{x}) . \quad (70)$$

However, this is not the only possible choice. Physically, the deformation of the interface/boundary mimics a no-penetration boundary condition, meaning that no fluid is allowed to

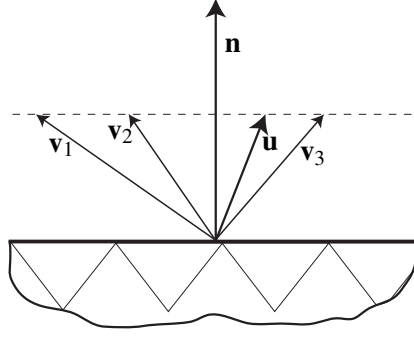


Figure 21: Possible displacement directions satisfying the kinematic boundary condition.

cross the interface – this condition will always be fulfilled if the interface moves in accordance with the fluid as in Equation (70). However, it will also be fulfilled by any choice where the mass flux through the boundary, \dot{m} , is 0. The mass flux \dot{m} through the interface can be computed as [75]:

$$\dot{m} = \int_{\Gamma^{int}} \rho(\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} \, d\mathbf{x} \stackrel{!}{=} 0. \quad (71)$$

Consequently, all choices for \mathbf{v} that comply with the kinematic boundary condition

$$\mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = \mathbf{u}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) \quad (72)$$

are valid (cf. Figure 21). It can be deduced from Equation (72), that tangential node movement remains irrelevant to the interface/boundary shape, but may significantly influence the mesh quality. Particularly applications with large tangential velocities – such as die swell in extrusion processes or rising droplets – profit tremendously if the tangential velocity component is modified or even suppressed when computing the interface/boundary velocity \mathbf{v} .

Note, however, that the assumptions made by the kinematic boundary condition are only valid strictly for the analytical geometry of the interface/boundary. This is particularly due to the computation of the normal vector, which is usually ambiguous at the finite element nodes (usual finite element discretizations employ shape functions, which offer only C^0 continuity across element boundaries). Still, the finite element nodes are exactly the points of the interface/boundary where the mesh velocity needs to be defined. It is therefore inherent to the nature of the discretization that Equation (72) can only be fulfilled approximately by any choice other than Equation (70). [202]

Nevertheless, Behr [21] indicates several possibilities for boundary displacement resulting from Equation (72): displacement with the normal velocity component with $\mathbf{v} = (\mathbf{u} \cdot \mathbf{n})\mathbf{n}$ and displacement only in a specific coordinate direction (e.g., y-direction), i.e., $\mathbf{v} = \frac{(\mathbf{u} \cdot \mathbf{n})\mathbf{e}_y}{\mathbf{n} \cdot \mathbf{e}_y}$. Note here that a direct projection onto the unit vector \mathbf{e}_y in the coordinate direction perpendicular to the main flow direction is not possible, as this would result in a displacement in the correct direction, but not with the correct magnitude to satisfy the kinematic boundary condition.

In cases with very extensive deformations, even this method can lead to strongly distorted elements. As a remedy, it is possible to allow the nodes to slip along the tangential direction. The amount of tangential movement is then not connected to the fluid velocity, but determined according to the same method responsible for the inner nodes, which will be described in the next section.

8.2.2 Retaining mesh quality: mesh update for inner nodes

As described in the previous section, both the ALE approach and the deforming domain space-time approach prescribe only the mesh displacement on (or possibly in the vicinity of) the boundary/interface. All other nodes have to be adapted to this displacement in a way that retains good mesh quality. Wall [216] identifies two possible categories for the treatment of the inner nodes: methods that rely on explicit functions, which have been defined a priori, and methods that require the solution of their own system of equations.

Examples of the first category can be found in [182, 126], where the boundary deformation is distributed smoothly throughout the entire domain using an interpolation kernel.

In the second category, probably the most popular option is the Elastic Mesh Update Method (EMUM) [122] and comparable approaches (e.g., [152, 166]). In this method, the computational mesh is treated as an elastic body reacting to the boundary deformation applied to it. For this purpose, the linear elasticity equation is solved for the mesh displacement \mathbf{v} , which relates to the mesh velocity \mathbf{v} as $\mathbf{v} = \mathbf{v}\Delta t$:

$$\nabla \cdot \boldsymbol{\sigma}_{\text{mesh}} = 0, \quad (73)$$

$$\boldsymbol{\sigma}_{\text{mesh}}(\mathbf{v}) = \lambda_{\text{mesh}} (\text{tr} \boldsymbol{\epsilon}_{\text{mesh}}(\mathbf{v})) \mathbf{I} + 2\mu_{\text{mesh}} \boldsymbol{\epsilon}_{\text{mesh}}(\mathbf{v}), \quad (74)$$

$$\boldsymbol{\epsilon}_{\text{mesh}}(\mathbf{v}) = \frac{1}{2} (\nabla \mathbf{v} + (\nabla \mathbf{v})^T). \quad (75)$$

In structural mechanics, λ_{mesh} and μ_{mesh} are the Lamé-parameters that are needed to define the stiffness tensor for isotropic and linear elastic materials. In the context of the elastic mesh update, these parameters are prescribed for each element in order to control its stiffness. This is used to increase the stiffness of the smaller elements compared to the larger elements in order to allow for larger deformation before the mesh becomes invalid. The choice of Lamé-parameters can differ significantly between the approaches.

As boundary conditions, the displacement of the boundary/interface nodes, which is prescribed due to the rules given in Section 8.2.1, is imposed:

$$\mathbf{v} = \hat{\mathbf{v}} \quad \text{on } \Gamma_{\mathbf{v}}(t). \quad (76)$$

Another possible boundary condition, which, in the context of mesh deformation, is usually used in addition to the condition above, would be:

$$\mathbf{n} \cdot \boldsymbol{\sigma}_{\text{mesh}} = \mathbf{h}_{\text{mesh}} \quad \text{on } \Gamma_{\mathbf{h}_{\text{mesh}}}(t). \quad (77)$$

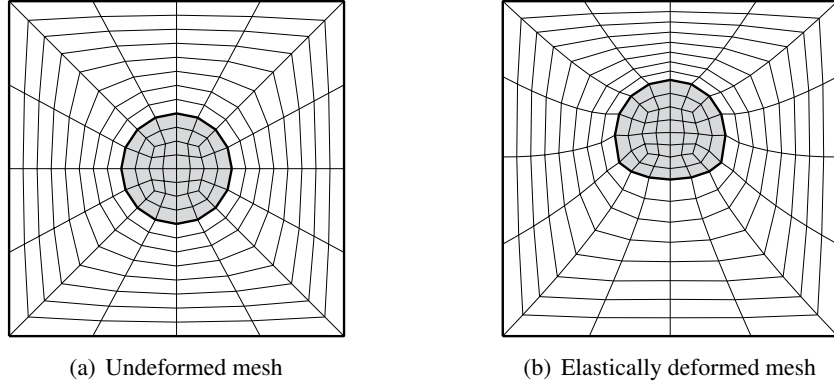


Figure 22: Elastic Mesh Update Method: An initially circular interface (a) is subject to a non-uniform motion. (b) shows the ideal elastic response of the entire mesh with respect to the interface movement.

This condition lets the nodes slip along certain boundaries. $\Gamma_{\mathbf{v}}(t)$ and $\Gamma_{\mathbf{h}_{\text{mesh}}}(t)$ are no longer distinct subsets of $\Gamma^{\text{int}}(t)$, but can be combined for different degrees of freedom. The calculated displacement \mathbf{v} can now be used to update the nodal coordinates in all phases:

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{v}. \quad (78)$$

As a possible alternative, a simple Laplace equation can be solved for the displacement instead of Equation (73), which diffuses the displacement throughout the domain, but gives less control over the individual elements.

Also in the second category, but from a completely different view point, a method where a functional associated with the mesh distortion is minimized is proposed in [141]. The approach presented in [130] is based on a series of optimization steps with regard to the mesh quality. Loubère et al. [144] apply a method, where all nodes are displaced in a fully Lagrangian fashion, but each time-step contains a reconnection step, where a new connectivity is established based on the existing nodal positions.

As a final option, remeshing at every time-step can be applied, as for example in [167] in the context of the particle finite element method (PFEM). Strictly speaking PFEM is a particle method, with marker particles carrying the information of the physical properties. What relates this method to the interface-tracking approach however is that the particles are connected by a Lagrangian mesh, where the marker particles simultaneously act as finite element nodes, on which the flow equations are evaluated.

8.3 Normal vector and curvature approximation

In principle, the explicit description of the interface/boundary allows the immediate evaluation of normals and curvature within the elements, as long as the shape functions can be differentiated sufficiently often. If the values for the normal vector or the curvature are needed directly at

element nodes, they can be obtained by a weighted average of the values within neighbouring elements. One example for defining this average can be found in [72].

In cases, where the geometric interpolation cannot provide the necessary geometric information – i.e., mainly in the case of linear interpolation –, one may either resort to the Laplace-Beltrami technique (cf. Section 3.2.1) or introduce an additional geometrical entity from which the desired values can be obtained. Here, Mier-Torrecilla et al. [158] propose the use of the osculating circle to the curve at each grid point. In the implementation, the osculating circle is identified as the circle with shoulder points at the respective node and its two neighbours. Using the radius r of the osculating circle, the term $\kappa \mathbf{n}$ can be evaluated as $\frac{r}{|r|^2}$. The use of spline interpolations for this purpose is proposed in [82] with the use of piecewise defined cubic splines, and in [69] with the use of a single NURBS curve or surface to represent the boundary.

The utilization of splines to obtain geometric quantities in interface tracking naturally leads to methods, which integrate Computer-Aided-Design (CAD) information into the finite element analysis.

8.4 Recent advancement in CAD-integration for the finite element method

Geometries for engineering applications are generated using Computer-Aided-Design (CAD) systems: the CAD model is what we assume to be the real geometry. As of now, all major CAD systems share one common basis for geometry representation: Non-Uniform Rational B-Splines (NURBS). They provide a common standard to exchange geometry information. Similar to the close relation between level-set methods and XFEM, an interconnection between CAD-related numerical methods and interface tracking can be identified. (Note however that in a few cases, the level-set method has also been combined with spline descriptions, as for example in [27, 81].)

In the classical discretization methods, the exact NURBS geometry is lost as a finite element mesh, which is in general only an approximation of the geometry, is generated. In this section, we will explain the structure of NURBS and then introduce two methods that make use of the NURBS format in order to integrate the exact geometry into the finite element method, thus avoiding the approximation character of the finite element mesh.

8.4.1 Geometry representation using splines

We will start out with the most basic NURBS structure: the NURBS curve. From the curve definition, it will later be straightforward to derive the definitions of higher-dimensional objects. This section is based on [189, 178]. NURBS are an example of a parametric geometry representation. In this type of representation, a local parameter (e.g., θ) is defined along the curve. Usually, we use $\theta = 0$ at one end of the curve and $\theta = 1$ at the other end. The global coordinates of the curve (x, y, z) are then each defined as functions of θ :

$$x = x(\theta), \quad y = y(\theta), \quad z = z(\theta). \quad (79)$$

Important specifications during the historical development of NURBS were to generate a curve

- (1) whose smoothness is completely under user control,

- (2) which occupies a predefined space,
- (3) which allows for local shape control, and
- (4) which is able to represent both free-forms and analytical shapes.

In the NURBS context, the space occupied by the curve is defined using control points, which are connected to a control polygon. It is a property of the NURBS that it will always remain within the convex hull of the control points (Requirement (2)). By modifying the position of the individual control points, the shape of the curve will also be modified, as the control points guide the curve. The NURBS definition allows giving certain control points a larger influence on the curve compared to others, as each control point is assigned a weight. Another important feature of a NURBS is that each control point is only responsible for the shape of a minor portion of the curve (Requirement (3)). In the NURBS definition, this effect is realized through the B-spline basis functions M , the building blocks of the NURBS basis which will be introduced below. The B-spline basis functions are non-zero only in a sub-domain of the parameter space. Each control point i is associated with exactly one basis function $M_{i,p}$ of polynomial degree p . It is the non-zero domain of this basis function that indicates which portion of the curve is influenced by the specific control point. This indication is given in terms of the local parameter θ . A typical basis function for a NURBS is depicted in Figure 23. The control point associated with this basis function will only influence curve points associated with $\theta_1 < \theta < \theta_4$. The values of θ , where the influence of a basis function begins and ends, are referred to as *knots*.

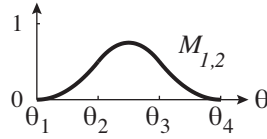


Figure 23: Example of a B-spline basis function with polynomial degree 2. $M_{1,2}$ is non-zero between θ_1 and θ_4 .

The curve definition \mathbf{C} of a NURBS curve is then

$$\mathbf{C}(\theta) = \sum_{i=1}^n R_{i,p}(\theta) \mathbf{P}_i, \quad (80)$$

where R_i denote the NURBS basis functions and \mathbf{P}_i are the coordinates of the control point i – which are always given in the global coordinate system (x, y, z) . n indicates the total number of control points. As we can see, \mathbf{C} is a mapping from the parametric coordinate θ to the global coordinate system (x, y, z) . Notice that a NURBS curve has only one local parameter, no matter if it exists in \mathbb{R}^1 or whether it is embedded into the \mathbb{R}^2 or the \mathbb{R}^3 . In this respect, NURBS are an example of an immersion.

The NURBS basis functions R emanate from the B-spline basis functions M . These in turn are usually defined recursively (cf. Figure 24), starting from the constant basis functions, using the Cox-deBoor relation:

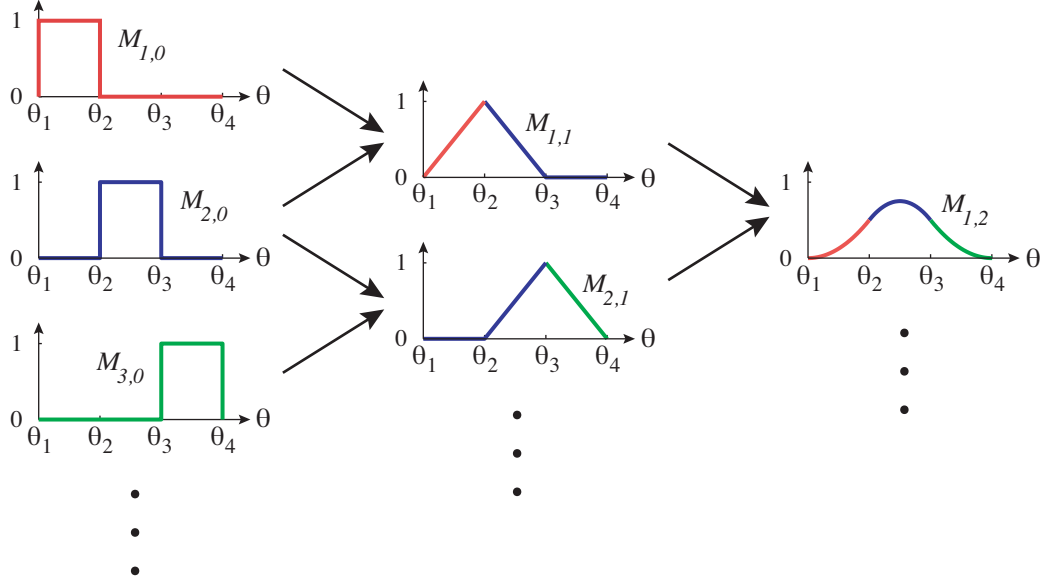


Figure 24: Recursive definition of the B-spline basis functions: For each degree p , the total number of basis functions reduces by one. Each basis function of the higher degrees is built-up of $p + 1$ constant basis functions, meaning that it is non-zero exactly between θ_i and θ_{i+p+1} . In any interval $\theta_i < \theta < \theta_{i+1}$, exactly $p + 1$ basis functions will be non-zero.

$$M_{i,0}(\theta) = \begin{cases} 1, & \theta_i \leq \theta < \theta_{i+1}, \\ 0, & \text{otherwise,} \end{cases} \quad (81)$$

$$M_{i,p}(\theta) = \frac{\theta - \theta_i}{\theta_{i+p} - \theta_i} M_{i,p-1}(\theta) + \frac{\theta_{i+p+1} - \theta}{\theta_{i+p+1} - \theta_{i+1}} M_{i+1,p-1}(\theta). \quad (82)$$

R is then defined as:

$$R_{i,p}(\theta) = \frac{M_{i,p}(\theta)w_i}{\sum_{i=1}^n M_{i,p}(\theta)w_i}. \quad (83)$$

The rational structure of the basis is responsible for the fact that NURBS are able to represent all analytical shapes, including conic sections (Requirement (4)). The values θ_i, θ_{i+p} , etc. are the knot values, i.e., the indication of the influence domain of the basis functions. The knot values are collected in the so-called knot vector:

$$\Theta = [\underbrace{\theta_1, \theta_2}_{M_{1,0} \neq 0}, \underbrace{\theta_3, \theta_4}_{M_{3,0} \neq 0}, \dots]. \quad (84)$$

For the proper definition of the basis, it is essential that the knots are in a non-decreasing sequence. However, it is specifically possible to repeat knot values. With each repetition, the continuity of the basis is decreased. With regard to the curve, there are then two measures that influence the continuity: the continuity of the basis and the topology of the control points. If control points are placed on top of each other, the curve continuity is decreased, e.g., to represent sharp corners. If control points are aligned, the continuity can be increased. Both effects are a direct consequence of property that the NURBS curve is always contained within the convex hull of its control polygon. These two measures give complete control over the curve continuity (Requirement (1)). In general, the basis functions, and therefore the curve, are infinitely differentiable at all points, and $p - 1$ times differentiable at the knots.

The derivatives and the curvature of NURBS curves can be calculated analytically. From differential geometry, the curvature of a parametric curve $\mathbf{C}(\theta)$ can be calculated with [94]:

$$\kappa(\theta) = \frac{|\mathbf{C}'(\theta) \times \mathbf{C}''(\theta)|}{|\mathbf{C}'(\theta)|^3}. \quad (85)$$

$\mathbf{C}'(\theta)$ and $\mathbf{C}''(\theta)$ denote the first and second derivative of the curve with respect to parameter θ . For the sake of brevity, $M_{i,p}(\theta)$ is written as $M_{i,p}$ and $\sum_{i=1}^n$ as \sum_i in the following equations.

$$\mathbf{C}'(\theta) = \frac{\sum_i M'_{i,p} w_i \mathbf{P}_i \sum_i M_{i,p} w_i - \sum_i M_{i,p} w_i \mathbf{P}_i \sum_i M'_{i,p} w_i}{(\sum_i M_{i,p} w_i)^2}, \quad (86)$$

$$\begin{aligned} \mathbf{C}''(\theta) = & \frac{\sum_i M''_{i,p} w_i \mathbf{P}_i \sum_i M_{i,p} w_i - \sum_i M_{i,p} w_i \mathbf{P}_i \sum_i M''_{i,p} w_i}{(\sum_i M_{i,p} w_i)^2} \\ & - \frac{2 \sum_i M'_{i,p} w_i \left(\sum_i M'_{i,p} w_i \mathbf{P}_i \sum_i M_{i,p} w_i \right)}{(\sum_i M_{i,p} w_i)^3} \\ & - \frac{2 \sum_i M'_{i,p} w_i \left(\sum_i M_{i,p} w_i \mathbf{P}_i \sum_i M'_{i,p} w_i \right)}{(\sum_i M_{i,p} w_i)^3}. \end{aligned} \quad (87)$$

$M'_{i,p}$ and $M''_{i,p}$ are the first and second derivative of the basis functions with respect to the parameter θ .

Figure 25 gives an example of a NURBS curve representing a circle. A closed curve is obtained by overlapping the first and the last control point. Higher continuity at the connection point of the curve can be reached if p control points at the ends are overlapped ($\mathbf{P}_{n-p+1} = \mathbf{P}_1, \dots, \mathbf{P}_n = \mathbf{P}_p$).

If surfaces (or volumes) are to be described through NURBS, a second (or third) local parameter is introduced. The basis functions for the higher-dimensional objects are computed as a Cartesian product of the univariate basis functions $M_{i,p}(\theta)$ and $L_{j,q}(\iota)$ defined in Equation (82). For each local coordinate direction, an individual knot vector is required. Note that it is possible to choose different polynomial degrees of the basis for each local coordinate. The NURBS surface \mathbf{S} is described as

$$\mathbf{S}(\theta, \iota) = \sum_{i=1}^n \sum_{j=1}^m R_{i,j}^{p,q}(\theta, \iota) \mathbf{P}_{i,j}, \quad (88)$$

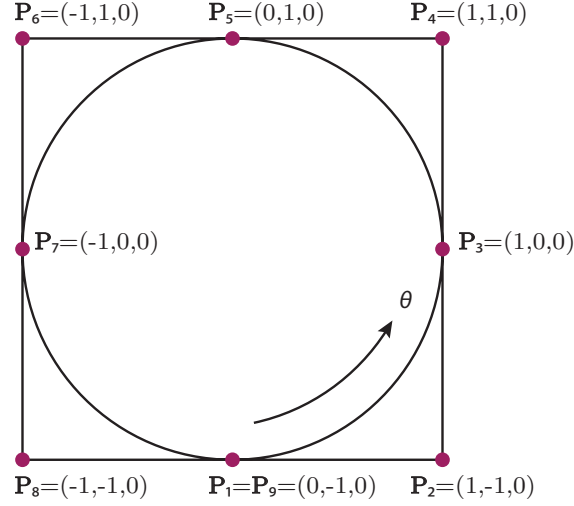


Figure 25: Example of a NURBS curve: with 9 control points and a quadratic basis, it is possible to represent a full circle. The corresponding knot vector is $\Theta = [0 \ 0 \ 0 \ 1/4 \ 1/4 \ 1/2 \ 1/2 \ 3/4 \ 3/4 \ 1 \ 1]$. The weights have to be set to $w_1 = w_3 = w_5 = w_7 = w_9 = 1$ and $w_2 = w_4 = w_6 = w_8 = \frac{\sqrt{2}}{2}$.

with

$$R_{i,j}^{p,q}(\theta, \iota) = \frac{M_{i,p}(\theta) L_{j,q}(\iota) w_{i,j}}{\sum_{\hat{i}=1}^n \sum_{\hat{j}=1}^m M_{\hat{i},p}(\theta) L_{\hat{j},q}(\iota) w_{\hat{i},\hat{j}}}. \quad (89)$$

8.4.2 Isogeometric analysis

Recall that the finite element method invokes the *isoparametric concept*, as described in [113] or many other works on finite elements. The isoparametric concept signifies that the element nodes are interpolated with the same shape functions as the unknown function, i.e., the solution of the partial differential equation. This approach instantly presents a mapping from reference coordinates to global coordinates, thus offering a simple means for evaluating all components of the transformation theorem necessary to compute the integrals of the weak form on a reference element instead of the global elements. In comparison to an integral evaluation in global coordinates, the approach using a reference element makes easy use of the local support of the shape functions and furthermore avoids the more tedious and computationally inefficient definition of the shape functions in global coordinates.

Cottrell, Bazilevs and Hughes used the exact same underlying approach when devising the concept of isogeometric analysis [114, 50, 20]. In the spirit of the isoparametric concept, it utilizes the NURBS basis functions in order to represent both the geometry and the unknown solution. For the geometry, the already presented definition (80) or (88) is employed. The unknown function u is then approximated in exactly the same fashion as:

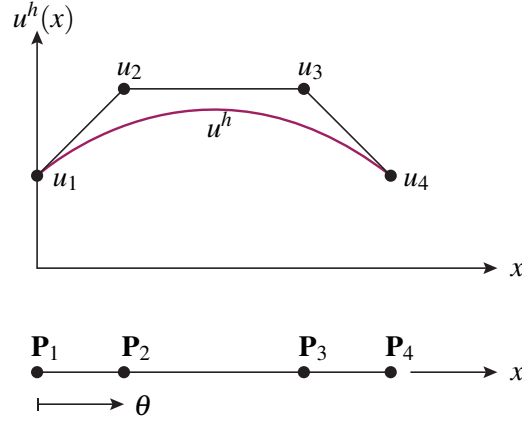


Figure 26: Function representation in IGA: Consider a quadratic NURBS curve whose control points are aligned in such a way that it represents a line (lower part of the picture). The local parameter along the curve is θ , and it is embedded into \mathbb{R}^1 with coordinate x . On this curve, a function u is approximated (u^h). The function is represented through the control variables u_1 through u_4 which are interpolated using the NURBS basis function $R_{i,p}$.

$$u^h = \sum_{i=1}^n R_{i,p}(\theta) u_i. \quad (90)$$

The values u_i are the control variables. These are the unknown values solved for in the finite element code. Note that just like the control points for the geometry, the values of the control variables are in general not coinciding with the solution; see Figure 26.

Not only does this idea essentially provide the capability to perform finite element analysis directly on CAD models, but it also profits from superior approximation properties of NURBS as compared to the polynomial shape functions used in the standard finite element method.

8.4.3 The NURBS-enhanced finite element method

One major challenge still remaining for the isogeometric analysis described in the previous section is the generation of closed volume splines describing complex geometries. In the traditional CAD/CAM systems, complex geometries are usually described as a large number of individual surface patches, connected neither geometrically nor parametrically. The reason for this is that traditionally, the CAD system tries to mimic classical manufacturing approaches (such as turning, drilling, or milling) starting out from a form similar to a semimanufactured product. This makes the handling of the CAD tool easily accessible to engineers. However, this also makes it very complicated to use for isogeometric analysis.

Sevilla, Fernandez-Mendes, and Huerta have proposed an alternative on middle ground between isogeometric analysis and standard finite elements: the NURBS-enhanced finite element method (NEFEM) [194, 196, 195, 202]. In NEFEM the boundary of the geometry is represented using

NURBS, thus leading to an exact representation, whereas for the interior, a standard finite element mesh, with all benefits of existing meshing algorithms, is utilized. Note that this leads to two different kinds of elements: (1) standard finite elements in the interior and (2) elements with one NURBS edge alongside the boundary. Usually, elements of category (1) will be in the vast majority, keeping the computation very efficient. Through the elements of category (2), the exact geometry is made available during the process of evaluating the integrals of the appropriate weak form (e.g., of the governing equations Equations (1)–(2)): The integration domain Ω^h is no longer only an approximation of the real domain Ω . The availability of the exact geometry is however not resorted to for the interpolation of the unknown solution. The latter is approximated using standard Lagrangian shape functions both along the boundary and in the interior. As a consequence, NEFEM is no longer an isoparametric method. This is in contrast to IGA.

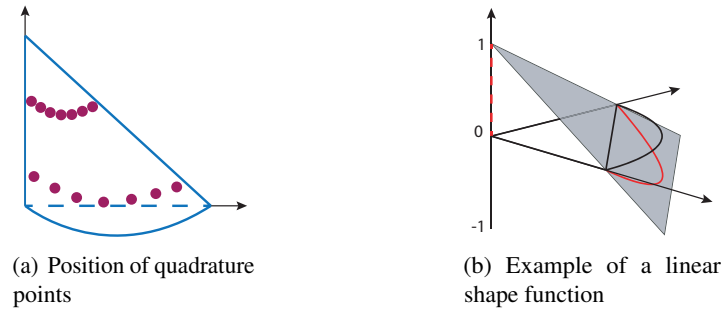


Figure 27: (a) The NEFEM quadrature points are adapted to the curved triangle shape. (b) Example of a linear shape function in the NEFEM context. Negative values and values larger than 1 may occur.

The main advantage of NEFEM over standard finite elements is that the position of the integration points needed in the finite element method are determined from the curved NURBS geometry and not from the approximated geometry (cf. Figure 27). If we consider boundary integral, such as the integral arising from the surface tension, the integration points are evenly distributed on the curved geometry. Due to the continued approximation of the unknown function through Lagrange polynomials an oddity arises: negative shape function values may occur along the boundary (cf. Figure 27). Furthermore, the shape function connected to all nodes, also those not located on the NURBS edge, are non-zero along the NURBS edge. This has the consequence, that interior nodes contribute to boundary integrals – a highly unusual situation in the finite element method.

This section concludes the introduction of the numerical methods. The next sections will concentrate on standard applications.

9 Bubbles and drops

The simulation of drops (liquid) or bubbles (gas) in a surrounding bulk phase is one of the classical benchmark cases in multi-phase flow. It addresses the issues of surface tension effects, moving interfaces, as well as the question of break-up and coalescence of phase domains (in

this case, the drops). The test case is motivated by the application of liquid-liquid or liquid-gas extraction columns [28]. The relevant effects can however also be extended for example to the simulation of the alveoli in the lung.

9.1 Static drop inside a rectangular domain

The first test case regularly used in the context of drop simulations is a circular drop inside a rectangular domain (cf. Figure 28(a)). The domain Ω_2 is occupied by fluid 2, which is surrounded by another fluid 1 residing in the domain Ω_1 . The dimensions of the domains differ within the literature. What remains in common is that the steady Stokes equations without external forces (i.e., in particular without gravity) are solved on the two domains (cf. Equation (5)–(6)). Density and viscosity of the two fluids are usually chosen to be equal. However, the presence of surface tension with a surface-tension coefficient of γ is assumed. Since the analytical value of the sum of principal curvatures is known for circles ($\kappa = \frac{1}{r}$) and spheres ($\kappa = \frac{2}{r}$), according to the Laplace-Young equations, the analytical solution is (cf. Section 2.1):

$$\mathbf{u}(\mathbf{x}) = \mathbf{0}, \quad (91)$$

$$p(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \in \Omega_1, \\ \gamma\kappa, & \mathbf{x} \in \Omega_2. \end{cases} \quad (92)$$

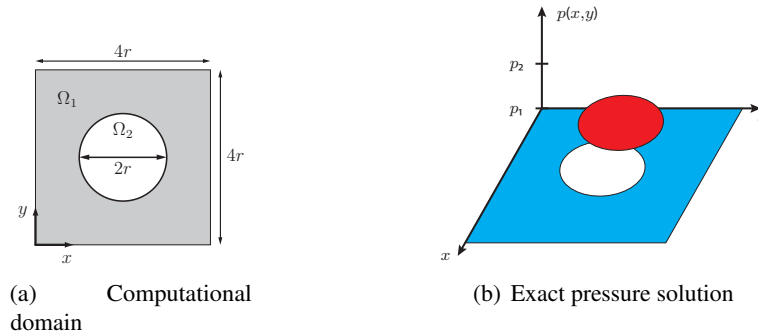


Figure 28: Static test case with circular interface. In (b), a zero reference pressure is assumed for Ω_1 . Furthermore, the parameters have been chosen to $\rho_1=\rho_2= 1.0kg/m^3$, $\mu_1=\mu_2= 1.0kg/m/s$, $\gamma = 1.0kg/s^2$ and $r = 0.5m$.

With the analytical solution at hand, this test case is particularly valuable to demonstrate the capabilities of a chosen method to handle the jump of quantities across the interface as well as the evaluation of geometrical quantities of the interface. This is even more the case as Ganesan et al. [86] point out that this solution is in general not represented well by a discretization scheme: the discrete velocities are not equal to $\mathbf{0}$ – we see spurious velocities. [86] identifies two causes: inadequate representation of the pressure solution (particularly the pressure jump) and an insufficient approximation of the curvature. They give the error bound on \mathbf{u}^h as (in our own notation):

$$|u^h|_1 \leq C \left(\inf_{q^h \in Q^h} \|p - q^h\|_0 + \sup_{w^h \in V^h} \frac{|(\kappa^h, w^h \cdot \mathbf{n}) - (\kappa, w^h \cdot \mathbf{n})|}{|w^h|_1} \right). \quad (93)$$

From theoretical considerations, Ganesan et al. [86] conclude that the use of continuous approximation functions in the case of a discontinuous solution is to always be avoided – this holds true even if the discontinuities are resolved by the mesh. The jump is then spread over several elements, thus distorting the solution (cf. Figure 29(a)). Here, and also in Gross and Reusken [97], it is shown that in general, the error in the pressure approximation $\inf_{q^h \in Q^h} \|p - q^h\|_0$ is bounded by $c\sqrt{h}$, leading to very slow convergence.

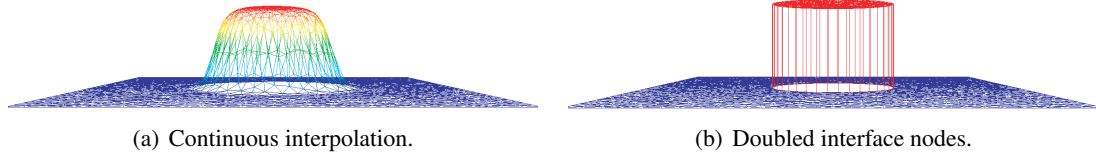


Figure 29: Comparison of the pressure jump using continuous interpolation functions and interface nodes with doubled pressure degrees of freedom [69]. ©Wiley-Blackwell

These theoretical results brought about a whole set of ideas aiming at numerically obtaining both an instant pressure jump and an improved curvature approximation.

Pressure jump: In the area of level-set methods, the work concentrated on developing problem-suitable finite element pressure spaces: XFEM evolved as the main means of incorporating the pressure jump (cf. Section 6.2). In [97, 95, 186], a Heaviside enrichment is employed for this purpose. In the case of continuous basis functions, convergence orders for the pressure of only $O(h^{3/2})$ in the L^2 -norm were reached. For the velocity, the reported convergence order is $O(h^{1/2})$ in the H^1 -norm. However, if the Heaviside-enrichment was employed in combination with an appropriate curvature approximation, a convergence order of $O(h^\alpha)$ with $\alpha \geq 1$ could be recovered. Also in the area of level-set approximations, Ausas et al. [11] present an approach where the pressure space is locally enriched at those elements, which contain the interface. The corresponding elements are subdivided along the interface, and each part of the domain is then only influenced by pressure degrees of freedom on one side of the interface. The method does not introduce new degrees of freedom for the pressure. In [201], the stability of this method is compared to the method of Coppola-Owen [48] already mentioned in Section 6.

Along similar lines, but in the context of the PFEM method, the pressure degrees of freedom at the interface are duplicated, once associated to Ω_1 and once to Ω_2 , in [158]. This leads to the possibility of an exact representation of the pressure jump even on coarse meshes. In the context of boundary-conforming meshes, the same idea has been employed in [69]. The velocity values of the doubled nodes are coupled in the context of the boundary condition in Equation (10), but the pressure is allowed to take on different values. A sample result is depicted in Figure 29. The advantage of the local approaches is that they take features of the individual problem into account, e.g., the knowledge that the pressure jump will only ever occur at the interface. This makes the enrichment less general, but usually much more efficient and less detrimental to the numerical scheme (such as, for example, the ill-conditioned matrices XFEM can induce).

Curvature approximation: Most approaches never compute the curvature of the interface, but resort to the Laplace-Beltrami technique or the continuum surface force technique as described in Section 3 (among many others [18]). Among the rare explicit approaches, several researchers have attempted the use of splines as an interface representation, from which the curvature can then be derived analytically. This includes [87] where piecewise interpolated cubic splines provide an additional description of the discretized interface. In [69, 202], a single NURBS (cf. Section 8.4) represents the entire circle, thus providing an excellent means for evaluating the curvature at any given point of the interface. [202] furthermore employs a space-time version of the NEFEM (cf. Section 8.4). Here, a combination of P1P1 finite elements and a quadratic NURBS for the geometry description is employed. With this combination, the analytical solution can be obtained exactly even on a coarse mesh with only 8 FE nodes on the interface.

9.2 Rising drops or bubbles

The test case described in the previous section can be extended when independent material parameters ($\mu_{1,2}$ and $\rho_{1,2}$) as well as gravity are taken into account. Depending on the density relation, the drop in Ω_2 will then either rise or fall in Ω_1 due to buoyancy. In comparison to the previous test case, the set-up of this section needs to additionally consider the velocity kink at the interface, the jump in the pressure gradient, and most of all, a scheme for deforming domains.

Experimentally, the phenomenon of a rising drop is for example described in [45]. The book defines three dimensionless numbers, any two of which completely characterize the flow regime. They involve the gravity g , the drop diameter d , viscosity μ and density ρ , as well as the surface-tension coefficient γ :

- The Reynolds number, relating inertia to friction forces,

$$Re = \frac{\rho_2 \sqrt{g d}}{\mu_2}, \quad (94)$$

- the Eötvös number, relating buoyancy forces and surface tension,

$$Eo = \frac{\rho_2 d^2 g}{\gamma}, \quad (95)$$

- and the Morton number, relating viscous forces and the surface tension,

$$Mo = \frac{g \mu_2^4}{\rho_2 \gamma^3} = \frac{Eo^3}{Re^4}. \quad (96)$$

Depending on these characteristic numbers, three regimes can be distinguished: spherical, ellipsoidal, and spherical cap. Spherical drops occur at low Re and Eo . The ellipsoidal regime is located around relatively high Re and intermediate Eo , while both high Re and Eo yield a drop in the spherical cap regime.

Hysing et al. published benchmark computations for a single rising bubble [116]. It contains two benchmark cases, one in the ellipsoidal regime and one in the spherical cap regime, featuring

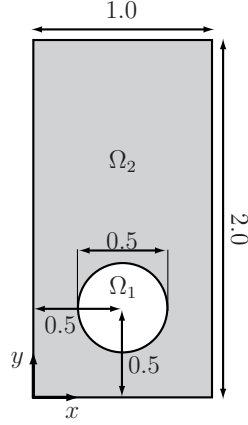


Figure 30: Rising bubble: Initial computational domain.

thin filaments and break-up. The results of three different numerical methods are described; two use the finite element method in conjunction with level-set and one finite elements in an ALE setting.

In [191], the first benchmark of [116] was repeated. Here, the surface tension effects dominate the bubble shape and no break-up should occur. The rectangular domain has the size of $1.0 \times 2.0m$ with an initially circular bubble with diameter $d = 0.5m$, as shown in Figure 30.

The properties of the fluids are $\rho_1 = 100kg/m^3$, $\rho_2 = 1000kg/m^3$, $\mu_1 = 1kg/m/s$, $\mu_2 = 10kg/m/s$, $f_y = -g = -0.98m/s^2$, and surface tension coefficient $\gamma = 24.5kg/s^2$. Under these circumstances, the characteristic Reynolds and Eötvös numbers can then be evaluated to:

$$Re = \frac{\rho_2 \sqrt{g d}}{\mu_2} = 35, \quad (97)$$

$$Eo = \frac{g \rho_2 d^2}{\gamma} = 10. \quad (98)$$

The spatial resolution is 80×160 elements and $\Delta t = 0.002s$. No-slip boundary conditions are assumed at the top and bottom boundary, slip boundary conditions are used along the vertical walls, and zero pressure is specified at the upper boundary. As an initial condition, the velocity field is set to $\mathbf{0}$. The results – Figure 31(a) reports the bubble shape over time – give further support to the conclusions in [116]. In Figure 31(a), the bubble shape at $t = 3.0s$ is compared with the result from [116], showing a very good agreement. In order to compare a time-evolving quantity, the rise velocity of the bubble, defined by the authors of the benchmark as:

$$v_{rise} = \frac{\int_{\Omega_1} v(\mathbf{x}, t) d\mathbf{x}}{\int_{\Omega_1} 1 d\mathbf{x}} \quad (99)$$

is depicted in Figure 31(b).

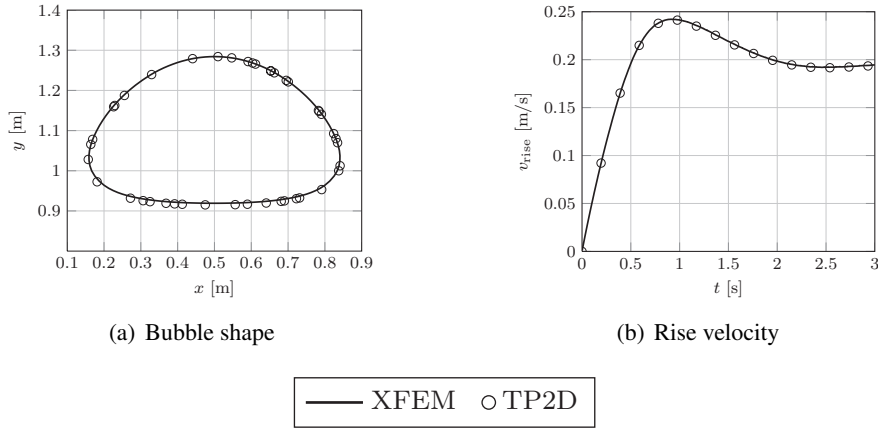


Figure 31: Rising bubble: Comparison of the bubble shape and rise velocity obtained with the XFEM level-set approach in [191] and simulation data from [116].

Other publications where this or comparable benchmarks have been studied are, e.g., the following. In [158], a particle finite element method (PFEM) is employed to evaluate the second test case. In this method, the marker particles are treated in a Lagrangian manner. The interaction between the particles is portrayed through a connecting mesh, which is renewed after each displacement update. In addition, the article discusses the difficulties in modeling break-up and coalescence, which are always mesh-dependent. Aland et al. consider both test cases with a phase field method [4]. The governing equations for the mixture of incompressible, immiscible fluids lead to a coupling of the momentum equation (1) with a Cahn-Hilliard type phase field equation. These are solved using an adaptive finite element scheme. Also with the phase-field method, rising bubble examples similar to [116] are investigated in [98]. A thermodynamically consistent set of governing equations is presented. The numerical scheme combines finite element discretizations for the momentum, phase-field, and chemical potential equations with a finite volume approach for all convective terms. Very large deformations as well as break-up and coalescence are inherent for these methods. In [129] the authors were able to reproduce both benchmark cases (with and without breakup) with the finite volume method. Qualitatively, the results show the same features as in [116]; the quantified results, however, present significant deviations from the ones obtained with the finite element codes. Doyeux et al. utilize the benchmark to validate their finite element method with interface capturing using level-set before they move on to the simulation of vesicles (a structure which serves as an imitation of the mechanical behavior of red blood cells). Again, both benchmark cases are presented [62]. In [3], the benchmark case is extended to 3D. The results of three different codes: XFEM with level-set, finite differences with level-set, and finite volumes with VOF are compared.

10 Sloshing tank

A second common application in free-surface flows are sloshing tanks. The numerical simulation of sloshing tank problems is motivated by, e.g., the analysis of storage tanks under seismic loading, or partially filled tanks on ships such as fuel tanks and maritime transport of fluids or liquid gas. The aim of the simulation is to gain information about the forces acting on the surrounding structures, e.g., the tank walls or the ship, in order to investigate possible failure mechanisms. When considering such effects, the phenomenon of sloshing is in general not negligible when investigating the responses of the tank structure to the load due to the large mass of the liquid in combination with the low stiffness of the tank walls [174]. If the problem is to be considered to its full extent, it falls into the category of fluid-structure interaction problems, where the thin structure of the tank wall reacts to the forces generated by the motion of the fluid and vice-versa. In this work, we will concentrate on the fluid component of such simulations.

The amplitude of the sloshing depends on the amplitude and frequency of the seismic load as well as the fluid fill level, the fluid properties, and the tank geometry [174]. In the next section, we will focus on the last point, including arbitrary tank geometries. Usually, the considered tanks are either rectangular or cylindrical. With regard to the fluid-structure-interaction problems, methods for arbitrarily shaped tanks need to be developed, however, in order to incorporate the different failure mechanisms of the tank wall into the sloshing phenomenon. Known shapes under failure of the tank structure are elephant footing, diamond-shaped buckling, as well as random convex and concave bumps [128].

Within the overall context of the fluid-structure-interaction problem, concessions are often made with regard to the accuracy of the fluid flow model. Many authors decide to model the liquid in the tank as inviscid, incompressible and irrotational. This enables the use of a simple Laplace equation, which describes the velocity potential. If the full Navier-Stokes equations (1)–(2) are chosen as governing equations, the sloshing tank is an application where the body forces are of great importance. In general, a gravitational force will be considered, since otherwise, any upward sloshing could never be reversed. In cases of rotating tanks, Coriolis forces are added to the general body forces. If seismic loading is to be considered, an alternating acceleration (usually orthogonal to gravity) is imposed.

Another factor, which categorizes different scenarios, is the regularity of the flow conditions. While moderate deformations can be accurately and efficiently handled by interface tracking methods, wave breaking, air entrapment, and touching of the tank ceiling is more easily conducted with interface-capturing methods. Benchmark problems in this area are in general restricted to rectangular tanks with either an initially flat surface, which is subsequently excited (e.g., [110, 208]), or a free surface with a prescribed original deformation, which is then allowed to settle into equilibrium (e.g., in [78, 131]). Even though this is a two-phase scenario, with a liquid phase in the container secluded by a gaseous phase on the top, in this application, the modelling is usually restricted to the liquid phase, as the density of the gaseous phase is several orders of magnitude lower [92]. Exceptions need to be made, e.g., if the flow conditions are so violent that gas entrapments occurs. Ansari et al. [8] use the modal method, a reduced-order method, to model tank sloshing based on the flow potential (Laplace equation). The authors investigate the question under which conditions the top fluid can be neglected during the simulation. The results were later further analyzed in [92]. In [88], the scope of the modal method is extended from

vertical walls to tapered walls. Numerical simulation and experimental validation of sloshing in a 2D rectangular tank is presented in [53]. The full Navier-Stokes equations are solved and the interface is captured using marker particles, which are updated in a Lagrangian manner. Subsequently, a global mass correction scheme is applied in order to ensure conservation of mass. Sloshing liquid motion in combination with a floating roof, which underlies the non-linear equation of motion, is the topic of [153]. The flow is modelled through the analytical solution modes of the velocity potential, and only the interface to the floating roof requires finite-element discretizations. In [174], the authors decide to solve the compressible Navier-Stokes equations with an ALE finite element method.

In a similar application, Lee et al. [133] simulate non-linear free-surface flows around ship bows including wave breaking phenomena. Due to the wave breaking, interface-capturing methods are more appropriate than interface tracking. In the referenced paper, a modified marker-density method is employed, circumventing the oversimplifications a traditional MAC-method would involve. As a solution method, finite differences with a sequential solution of velocity and pressure are used. One important challenge in this type of simulation is the conjunction point between the ship hull and the free surface, namely imposing the appropriate boundary conditions.

10.1 The boundary conditions

The free surface requires a no-penetration boundary condition (cf. Equation (70)), indicating that the shape of the free surface is dictated through the fluid velocity. If one would like to consider surface tension effects in the pressure solution, Equation (11) needs to be incorporated. Note that any technique relying on partial integration now needs to take the boundary integral into account (in contrast to the situation described in Section 3), as the free surface in a sloshing tank is not closed. An exception can be made if we assume that the contact angle θ , i.e., the angle between the tangent vector along Γ^{int} , $\hat{\mathbf{n}}'$, and the boundary Γ is assumed to be constant at 90° (cf. Figure 32). Then the boundary term is again 0 as shown in [84]. This generalization is acceptable if wetting effects, as described in [125], are assumed to not play an influential role. This holds true for the sloshing tank: As Behr points out in [22], the capillary effects are negligible in the case of a sloshing tank, thus precluding the need for a specific wetting model. Instead, a global slip condition is found appropriate to allow the free surface to slide freely along the tank walls. The slip boundary condition can be expressed in the following way [22]:

$$\begin{aligned} \mathbf{u} \cdot \mathbf{n} &= 0 & \text{on } \Gamma_{slip} \\ \mathbf{t}_1^T \boldsymbol{\sigma}(\mathbf{u}, p) \mathbf{n} &= 0 & \text{on } \Gamma_{slip} \\ \mathbf{t}_2^T \boldsymbol{\sigma}(\mathbf{u}, p) \mathbf{n} &= 0 & \text{on } \Gamma_{slip}. \end{aligned} \tag{100}$$

As an alternative, a Navier slip condition [197] could be employed to incorporate at least some amount of wall friction.

The tangential and normal vectors can either be computed analytically, if an appropriate description is available, or from the finite element mesh.

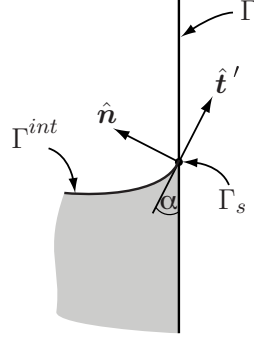


Figure 32: Close-up of the contact line Γ_s between an interface Γ^{int} and a domain boundary Γ . The tangent vector of the interface at Γ_s , $\hat{\mathbf{n}}'$, and the boundary Γ span the contact angle α . If wetting models are not considered, $\alpha = 90^\circ$.

10.2 Tanks with walls of arbitrary shape

If interface-capturing methods are used, the shape of the tank wall is insignificant, as long as the computed velocity field complies with the slip boundary condition. Since the marker particles, the level-set field, or the VOF field are advected with the fluid velocity, they will obey the shape of the boundary. In an interface-tracking context, tanks with walls of arbitrary shape have so far posed a significant challenge. The reasons for this are two-fold. One reason lies in the mesh deformation, which needs to comply with the analytic boundary at all times. This means that in addition to the discretized boundary, sufficient information on the analytic boundary needs to be stored, e.g., in form of a CAD model. The second reason is due to the discrepancy between the discrete slip condition and the analytic slip condition. In [22], it was found that this remains true even if a conforming normal vector is utilized.

The general procedure for incorporating a slip boundary condition into an interface-tracking context is to first assemble the full system matrix \mathbf{A} , irrespective of any boundary conditions that need to be applied. Subsequently, all equations are rotated in such a way, that they are now formulated in a coordinate system spanned by the tangent and the normal vector of the respective boundary. Only then are the Dirichlet conditions, e.g., zero velocity in normal direction, applied. This again requires two steps: (1) At any finite element node (or degree of freedom) with a Dirichlet boundary condition, the corresponding weighting function is equal to zero. Therefore, the corresponding equation is deleted from the matrix \mathbf{A} . (2) Furthermore, all occurrences of the specified degree of freedom are moved to the right-hand-side. Note that, usually, we encounter homogeneous Dirichlet conditions, where nothing needs to be done in this second step. The described procedure is straightforward if the tangential coordinate system is aligned with the main coordinate axes. In all other cases, the tangential coordinate system together with the corresponding rotation matrix remains to be determined.

Consider the original linear system of equations with the nodal velocities in the (x, y, z) -coordinate system:

$$\underbrace{\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \end{pmatrix}}_{\mathbf{u}} = \underbrace{\begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \end{pmatrix}}_{\mathbf{b}}. \quad (101)$$

For each node i with degrees of freedom $\mathbf{u}_i = (u_{i,x} \ u_{i,y} \ u_{i,z})^T$ a rotation matrix \mathbf{O}_i is determined, which transforms from the (x, y, z) -coordinate system to a given tangential coordinate system. Rotation matrices are square, orthogonal matrices with determinant 1 (otherwise $\mathbf{O}^{-1} = \mathbf{O}^T$ would not hold and the rotation could not be easily reversed). They can be obtained by placing the maps of the basis vectors of \mathbb{R}^{nsd} into the columns of the rotation matrix. The local rotation matrix then takes the form [47]:

$$\mathbf{O}_i = (\mathbf{t}_1 \ \mathbf{t}_2 \ \mathbf{n}). \quad (102)$$

With the help of the rotation matrix, new degrees of freedom $\tilde{\mathbf{u}}_i$ are defined in the tangential coordinate system:

$$\mathbf{u}_i = \mathbf{O}_i \tilde{\mathbf{u}}_i. \quad (103)$$

Equation (101) may then be formulated in terms of the new degrees of freedom:

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \mathbf{O}_1 & 0 & \dots \\ 0 & \mathbf{O}_2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{u}}_1 \\ \tilde{\mathbf{u}}_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \end{pmatrix}. \quad (104)$$

With

$$\begin{pmatrix} \mathbf{O}_1 & 0 & \dots \\ 0 & \mathbf{O}_2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{O}_1^{-1} & 0 & \dots \\ 0 & \mathbf{O}_2^{-1} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} = \begin{pmatrix} \mathbf{O}_1^T & 0 & \dots \\ 0 & \mathbf{O}_2^T & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}, \quad (105)$$

we can solve the system using:

$$\begin{pmatrix} \mathbf{O}_1^T & 0 & \dots \\ 0 & \mathbf{O}_2^T & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \mathbf{A} \begin{pmatrix} \mathbf{O}_1 & 0 & \dots \\ 0 & \mathbf{O}_2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{u}}_1 \\ \tilde{\mathbf{u}}_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{O}_1^T & 0 & \dots \\ 0 & \mathbf{O}_2^T & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \end{pmatrix}. \quad (106)$$

For analytical domain shapes (straight or tilted lines, circles), [22] already shows how the rotation matrix \mathbf{O} can be formed. In order to cope with arbitrary domain shapes, we propose a

boundary description that is based on NURBS (cf. Section 8.4 for the introduction to NURBS and the notation). The use of NURBS as a boundary description has the following advantages: (1) NURBS can represent arbitrary analytical and free-form shapes, (2) they can be extracted from CAD-models, (3) tangent and normal vectors can be analytically computed at any point on the NURBS. Due to point (3), the alignment of the system of equations of the flow solution can be performed in the standard way using a rotation matrix \mathbf{O} obtained as in Equation (102).

The first tangent vector is obtained as the first derivative of the NURBS surface (or curve) with respect to the first local parameter θ ; a second tangent vector can be obtained by taking the derivative with respect to the second local parameter ι :

$$\mathbf{t}_{1,NURBS} = \frac{\partial \mathbf{S}(\theta, \iota)}{\partial \theta}; \quad \mathbf{t}_{2,NURBS} = \frac{\partial \mathbf{S}(\theta, \iota)}{\partial \iota}. \quad (107)$$

Note, however, that in physical space, these two tangent vectors are not orthogonal on each other. This only holds for the parameter space spanned by θ and ι . Since the rotation matrix \mathbf{O} requires orthogonality in physical space, Gram-Schmidt orthogonalization is performed to obtain the two tangent vectors $\mathbf{t}_{1,n}$ and $\mathbf{t}_{2,n}$:

$$\mathbf{t}_{1,n} = \frac{\mathbf{t}_{1,NURBS}}{\|\mathbf{t}_{1,NURBS}\|}; \quad (108)$$

$$\mathbf{t}_{2,n} = \frac{\mathbf{t}_{2,NURBS} - (\mathbf{t}_{2,NURBS} \cdot \mathbf{t}_{1,n})\mathbf{t}_{1,n}}{\|\mathbf{t}_{2,NURBS} - (\mathbf{t}_{2,NURBS} \cdot \mathbf{t}_{1,n})\mathbf{t}_{1,n}\|}. \quad (109)$$

From the two tangent vectors, the normal vector can be computed via cross product.

Applying a similar procedure to the mesh deformation scheme is slightly more complex. The aim is that the boundary nodes of the mesh slide along the arbitrarily shaped boundary in order to accommodate the deformation imposed through the free surface. The individual displacement for each node \mathbf{v} is to be determined via the mesh deformation equation (73). However, as for the flow solution, we require a slip boundary condition. In addition to this condition, which restrict the nodal displacement to the local tangent direction, the requirement that the nodes remain on the NURBS at all times needs to be fulfilled. In total, the following steps are performed:

1. Rotate the mesh equation into the local tangential coordinate systems.
2. Drop the equation responsible for the normal component.
3. Compute the tangential displacement.
4. From the displacement in \mathbf{x} -coordinates, deduce a displacement on the NURBS in the local NURBS coordinates θ and ι .

The steps 1. through 3. are performed within the solution of the mesh deformation equation. Step 4. involves a transformation of the displacement given in tangential coordinates into the local NURBS coordinates. If the displacement is directly available in NURBS coordinates, it is ensured that all nodes remain on the NURBS. The following relation holds for the displacement

from one iteration (it) to the next ($it + 1$), where an iteration can be both a linearization step (such as in the Newton-Raphson algorithm) or a time step:

$$\mathbf{x}_{it+1} - \mathbf{x}_{it} = \mathbf{S}(\boldsymbol{\theta}_{it+1}) - \mathbf{S}(\boldsymbol{\theta}_{it}) \underbrace{\approx}_{\text{linearization}} \mathbf{J}\Delta\boldsymbol{\theta}. \quad (110)$$

$\Delta\boldsymbol{\theta}$ refers to the displacement in NURBS-coordinates and \mathbf{J} is the Jacobian of \mathbf{S} with respect to $\boldsymbol{\theta}$. From Equation (110), $\Delta\boldsymbol{\theta}$ can be computed. Figure 33 illustrates the resulting displacement.

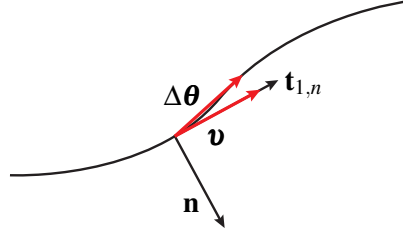


Figure 33: The nodal displacement \mathbf{v} is computed in terms of the tangential coordinate system spanned by $\mathbf{t}_{1,n}, \mathbf{t}_{2,n}, \mathbf{n}$. This displacement is transformed into a displacement expressed in the NURBS parametric coordinates $\boldsymbol{\theta}$ based on a local linearization of the NURBS curve.

In 2D, $\mathbf{J} = \frac{\partial \mathbf{C}}{\partial \boldsymbol{\theta}}$ and $\Delta\boldsymbol{\theta}$ can be readily obtained. In 3D, \mathbf{J}^{-1} cannot be determined easily and the (overdetermined) equation system needs to be solved. Starting point is step 3., where the tangential displacement $\Delta t_1, \Delta t_2$ has been obtained. The displacement reads

$$(\mathbf{t}_{1,n} \quad \mathbf{t}_{2,n}) \begin{pmatrix} \Delta t_1 \\ \Delta t_2 \end{pmatrix}. \quad (111)$$

This displacement can be exactly represented as a displacement in the $\mathbf{t}_1, \mathbf{t}_2$ -coordinate system – i.e., the coordinate system aligned with the NURBS local coordinates. What remains is to determine the appropriate magnitude of the displacement, $\Delta\boldsymbol{\theta}, \Delta\mathbf{t}$, which fulfils the relation

$$(\mathbf{t}_{1,n} \quad \mathbf{t}_{2,n}) \begin{pmatrix} \Delta t_1 \\ \Delta t_2 \end{pmatrix} = (\mathbf{t}_1 \quad \mathbf{t}_2) \begin{pmatrix} \Delta\boldsymbol{\theta} \\ \Delta\mathbf{t} \end{pmatrix}. \quad (112)$$

Note that $(\mathbf{t}_1 \quad \mathbf{t}_2)$ is also the Jacobian matrix \mathbf{J} . The above relation can be rewritten as

$$\Delta t_1 \mathbf{t}_{1,n} + \Delta t_2 \mathbf{t}_{2,n} = \Delta\boldsymbol{\theta} \mathbf{t}_1 + \Delta\mathbf{t} \mathbf{t}_2. \quad (113)$$

With the aim of comparing coefficients, the expressions (109) are inserted in Equation (113)

$$\Delta t_1 \frac{\mathbf{t}_1}{\|\mathbf{t}_1\|} + \Delta t_2 \frac{\mathbf{t}_2 - (\mathbf{t}_2 \cdot \mathbf{t}_{1,n}) \mathbf{t}_{1,n}}{\|\mathbf{t}_2 - (\mathbf{t}_2 \cdot \mathbf{t}_{1,n}) \mathbf{t}_{1,n}\|} = \Delta \theta \mathbf{t}_1 + \Delta t \mathbf{t}_2. \quad (114)$$

leading to the final result of

$$\Delta \theta = \frac{1}{\|\mathbf{t}_1\|} \left(\Delta t_1 - \frac{\Delta t_2 (\mathbf{t}_2 \cdot \mathbf{t}_{1,n})}{\|\mathbf{t}_2 - (\mathbf{t}_2 \cdot \mathbf{t}_{1,n}) \mathbf{t}_{1,n}\|} \right) \quad (115)$$

$$\Delta t = \frac{\Delta t_2}{\|\mathbf{t}_2 - (\mathbf{t}_2 \cdot \mathbf{t}_{1,n}) \mathbf{t}_{1,n}\|} \quad (116)$$

In such cases, one final remark needs to be made about the intersection between the free surface and the wall (Γ_s in Figure 32). Here, it is important that the displacement of the free surface is chosen compatible with the displacement along the wall.

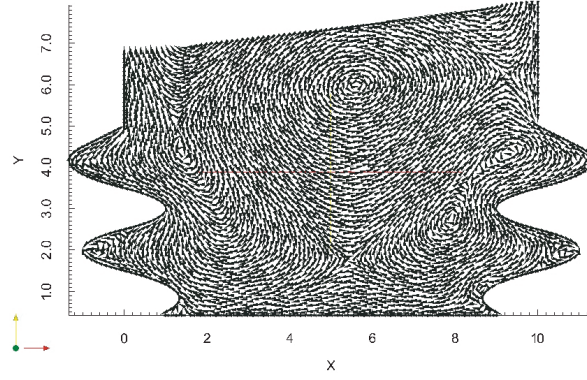
The effect of the new implementation of the slip condition is detailed by means of a numerical example.

In the test case, we consider an undeformable tank with curved side walls as indicated in Figure 34. The tank wall is defined through a quadratic NURBS curve with 18 control points. Along all walls, a slip boundary condition is imposed. The top boundary constitutes a free-surface. The tank is subjected to a downward gravitational acceleration $g = -0,98m/s^2$ and a horizontal sinusoidal acceleration $s = -0.2\sin(t)$. The considered fluid has a density of $\rho = 1000kg/m^3$ and a viscosity of $\mu = 0.1kg/m/s$. Figure 34 depicts the velocity vectors of the flow solution after $4s$ and $13.5s$. Note how the velocity aligns with the boundary. The mesh deformation is conform with the boundary at all times.

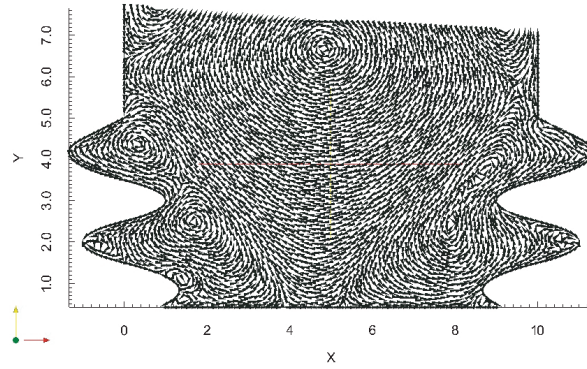
11 Conclusion

This paper gave an overview of the challenges in the simulation of fluid flow in deformable domains. Five main methods have been introduced with their individual advantages and drawbacks. As was seen, some may be used more often than others, but each has its niche. A recent area for the Marker and Cell method has been found in flow simulations for movies, where a qualitatively correct solution is more important than quantitative accuracy. Level-set has its main advantage when topological changes are to be expected within the domain. The volume-of-fluid method is utilized frequently in commercial codes. In cases where boundary conditions are difficult to impose, the use of the phase-field method can be worthwhile. The interface tracking approach may have a limited scope of applications, but comes with very high computational efficiency.

Out of a very large number of possibilities, the applications of drops and sloshing tanks were selected to demonstrate some of the approaches when computing free-boundary problems.



(a) $t = 4.0s$



(b) $t = 13.5s$

Figure 34: Sloshing in a tank with curved side walls: the tank wall is described by a quadratic NURBS curve with knot vector $[0 \ 0 \ 0 \ 1/16 \ 2/16 \ 3/16 \ 4/16 \ 5/16 \ 6/16 \ 7/16 \ 8/16 \ 9/16 \ 10/16 \ 11/16 \ 12/16 \ 13/16 \ 14/16 \ 15/16 \ 1 \ 1 \ 1]$ and control points $(10, 10); (10, 5); (10, 5); (12, 4); (8, 3); (12, 2); (8, 1); (10, 0); (10, 0); (0, 0); (0, 0); (2, 1); (-2, 2); (2, 3); (-2, 4); (0, 5); (0, 5); (0, 10)$. The velocity vectors of the flow solution are depicted after $4s$ and $13.5s$. Note how the velocity aligns with the boundary. [200]

Acknowledgements

The authors gratefully acknowledge support from the German Research Foundation (DFG) through the SFB 1120 “Thermal Precision”, the Emmy-Noether-research group “Numerical methods for discontinuities in continuum mechanics”, and the DFG program GSC 111 (AICES Graduate School). The computations were conducted on computing clusters provided by the Jülich Aachen Research Alliance (JARA). Furthermore, we would like to thank Philipp Knechtges for the infinite patience with which he shared his mathematical insight.

References

- [1] Numerical algorithms on a staggered grid. `irs.ub.rug.nl/dbi/43789c2bd41de`.
- [2] D. Adalsteinsson and J. Sethian. Transport and diffusion of material quantities on propagating interfaces via level set methods. *J. Comput. Phys.*, 185(1):271–288, 2003.
- [3] J. Adelsberger, P. Esser, M. Griebel, S. Groß, M. Klitz, and A. Rüttgers. 3d incompressible two-phase flow benchmark computations for rising droplets. Technical report, University of Bonn, 2014.
- [4] S. Aland and A. Voigt. Benchmark computations of diffuse interface models for two-dimensional bubble dynamics. *Int. J. Numer. Methods Fluids*, 69:747–761, 2012.
- [5] S. Aland, J. Lowengrub, and A. Voigt. Two-phase flow in complex geometries: A diffuse domain approach. *Comp. Mod. Engrg. Sci.*, 57(1):77–106, 2010.
- [6] A. Amsden and F. Harlow. A simplified MAC technique for incompressible fluid flow calculations. *J. Comput. Phys.*, 6(2):322–325, 1970.
- [7] W. Aniszewski, T. Ménard, and M. Marek. Volume of Fluid (VOF) type advection methods in two-phase flow: A comparative study. *Computers & Fluids*, 97:52–73, 2014.
- [8] M. Ansari, R. Firouz-Abadi, and M. Ghasemi. Two phase modal analysis of nonlinear sloshing in a rectangular container. *Ocean Engineering*, 38:1277–1282, 2011.
- [9] M. Arienti, X. Li, M. S. C. Eckett, M. Sussman, and R. Jensen. Coupled level-set/volume-of-fluid method for simulation of injector atomization. *Journal of Propulsion and Power*, 29(1):147–157, 2013.
- [10] F. Aubert, E. Aulisa, S. Manservigi, and R. Scardovelli. Interface tracking with dynamically-redistributed surface markers in unstructured quadrangular grids. *Computers & Fluids*, 35: 1332–1343, 2006.
- [11] R. Ausas, F. Sousa, and G. Buscaglia. An improved finite element space for discontinuous pressures. *Comp. Methods Appl. Mech. Engrg.*, 199(17):1019–1031, 2010.
- [12] O. Axelsson. *Iterative solution methods*. Cambridge University Press, Cambridge, 1994.

- [13] O. Axelsson, P. Boyanova, M. Kronbichler, M. Neytcheva, and X. Wu. Numerical and computational efficiency of solvers for two-phase problems. *Computers & Mathematics with Applications*, 65(3):301–314, 2013.
- [14] I. Babuska, J. Chandra, J. Flaherty, and U. S. A. R. O. M. S. Division. Adaptive computational methods for partial differential equations. In *Proceedings in Applied Mathematics Series*. Society for Industrial and Applied Mathematics, 1983.
- [15] M. Baltussen, J. Kuipers, and N. Deen. A critical comparison of surface tension models for the volume of fluid method. *Chemical Engineering Science*, 109:64–74, 2014.
- [16] E. Bänsch. Finite element discretization of the navier-stokes equations with a free capillary surface. *Numer. Math.*, 88(2):203–235, 2001.
- [17] R. Barreira, C. Elliott, and A. Madzvamuse. The surface finite element method for pattern formation on evolving biological surfaces. *Journal of Mathematical Biology*, 63(6):1095–1119, 2011.
- [18] J. Barrett, H. Garcke, and R. Nürnberg. Eliminating spurious velocities with a stable approximation of viscous incompressible two-phase stokes flow. *Comp. Methods Appl. Mech. Engrg.*, 267:511–530, 2013.
- [19] G. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 1967.
- [20] Y. Bazilevs, V. Calo, J. Cottrell, J. Evans, T. Hughes, S. Lipton, M. Scott, and T. Sederberg. Isogeometric analysis using T-splines. *Comp. Methods Appl. Mech. Engrg.*, 199:229–263, 2010.
- [21] M. Behr. *Stabilized Finite Element Methods for Incompressible Flows with Emphasis on Moving Boundaries and Interfaces*. PhD thesis, University of Minnesota, Department of Aerospace Engineering and Mechanics, 1992.
- [22] M. Behr. On the application of slip boundary condition on curved boundaries. *Int. J. Numer. Methods Fluids*, 45(1):43–51, 2004.
- [23] M. Behr, A. Johnson, J. Kennedy, S. Mittal, and T. Tezduyar. Computation of incompressible flows with implicit finite element implementations on the Connection Machine. *Comp. Methods Appl. Mech. Engrg.*, 108:99–118, 1993.
- [24] T. Belytschko and T. Black. Elastic crack growth in finite elements with minimal remeshing. *Internat. J. Numer. Methods Engrg.*, 45(5):601–620, 1999.
- [25] T. Belytschko, J. Kennedy, and D. Schoeberle. Quasi-eulerian finite element formulation for fluid-structure interaction. *Journal of Pressure Vessel Technology*, 102(1):62–69, 1980.
- [26] T. Belytschko, R. Gracie, and G. Ventura. A review of extended/generalized finite element methods for material modeling. *Modelling and Simulation in Materials Science and Engineering*, 17(4), 2009.

- [27] O. Bernard, D. Friboulet, P. Thévenaz, and M. Unser. Variational B-spline level-set method for fast image segmentation. In *Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008. 5th IEEE International Symposium on*, pages 177–180. IEEE, 2008.
- [28] E. Bertakis, S. Groß, J. Grande, O. Fortmeier, A. Reusken, and A. Pfennig. Validated simulation of droplet sedimentation with finite-element and level-set methods. *Chemical Engineering Science*, 65(6):2037–2051, 2010.
- [29] M. Bertalmio, L.-T. Cheng, S. Osher, and G. Sapiro. Variational problems and partial differential equations on implicit surfaces. *J. Comput. Phys.*, 174(2):759–780, 2001.
- [30] W. Boettinger, J. Warren, C. Beckermann, and A. Karma. Phase-Field Simulation of Solidification. *Annual Review of Materials Research*, 32:163–194, 2002.
- [31] R. Bonnerot and P. Jamet. Numerical computation of the free boundary for the two-dimensional Stefan problem by space-time finite elements. *J. Comput. Phys.*, 25:163–181, 1977.
- [32] J. Brackbill, D. Kothe, and C. Zemach. A continuum method for modeling surface tension. *J. Comput. Phys.*, 100:335–354, 1992.
- [33] A. Caboussat. Numerical simulation of two-phase free surface flows. *Archive Comp. Mech. Engrg.*, 12(2):165–24, 2005.
- [34] R. Chan and R. Street. A computer study of finite-amplitude water waves. *J. Comput. Phys.*, 6:68–94, 1970.
- [35] Y. Chang, T. Hou, B. Merriman, and S. Osher. A level set formulation of Eulerian interface capturing methods for incompressible fluid flows. *J. Comput. Phys.*, 124(2):449–464, 1996.
- [36] S. Chen, B. Merriman, S. Osher, and P. Smereka. A simple level set method for solving Stefan problems. *J. Comput. Phys.*, 135(1):8–29, 1997.
- [37] K. Cheng. *h- and p-XFEM with application to two-phase incompressible flow*. PhD thesis, RWTH Aachen University, 2010.
- [38] J. Chessa and T. Belytschko. An enriched finite element method and level sets for axisymmetric two-phase flow with surface tension. *Internat. J. Numer. Methods Engrg.*, 58(13):2041–2064, 2003.
- [39] J. Chessa and T. Belytschko. An extended finite element method for two-phase fluids. *J. Appl. Mech., ASME*, 70(1):10–17, 2003.
- [40] J. Chessa and T. Belytschko. Arbitrary discontinuities in space-time finite elements by level sets and X-FEM. *Internat. J. Numer. Methods Engrg.*, 61:2595–2614, 2004.
- [41] J. Chessa, P. Smolinski, and T. Belytschko. The extended finite element method (XFEM) for solidification problems. *Internat. J. Numer. Methods Engrg.*, 53(8):1959–1977, 2002.

- [42] S. Chippada, B. Ramaswamy, and M. Wheeler. Numerical simulation of hydraulic jump. *Internat. J. Numer. Methods Engrg.*, 37(8):1381–1397, 1994.
- [43] M. Cho, H. Choi, and J. Yoo. A direct reinitialization approach for level-set/splitting finite element method for simulating incompressible two-phase flows. *Int. J. Numer. Methods Fluids*, 67(11):1637–1654, 2010.
- [44] S. Choi, K. Lee, K. Hong, S. Shin, and O. Gudmestad. Nonlinear wave forces on an offshore wind turbine foundation in shallow waters. *International Journal of Ocean System Engineering*, 3(2):68–76, 2013.
- [45] R. Clift, J. Grace, and M. Weber. *Bubbles, Drops, and Particles*. Academic Press, 1978.
- [46] D. Cline, D. Cardon, and P. Egbert. Fluid flow for the rest of us: Tutorial of the marker and cell method in computer graphics. Technical report, Brigham Young University, 2013.
- [47] G. Collins. *The Foundations of Celestial Mechanics*. Pachart Publishing House, 2004.
- [48] A. Coppola-Owen and R. Codina. Improving Eulerian two-phase flow finite element approximation with discontinuous gradient pressure shape functions. *Int. J. Numer. Methods Fluids*, 49(12):1287–1304, 2005.
- [49] H. Coppola-Owen. *A Finite Element Method for Free Surface and Two Fluid Flows on Fixed Meshes*. PhD thesis, Universitat Politècnica de Catalunya, 2009.
- [50] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, Ltd, 2009.
- [51] A. Coutinho. Some reflections on big data and computational mechanics. *iacm expressions*, 35, 2014.
- [52] R. Croce, M. Griebel, and M. A. Schweitzer. Numerical simulation of bubble and droplet deformation by a level set approach with surface tension in three dimensions. *Int. J. Numer. Methods Fluids*, 62(9):963–993, 2010.
- [53] M. Cruchaga, R. Reinoso, M. Storti, D. Celentano, and T. Tezduyar. Finite element computation and experimental validation of sloshing in rectangular tanks. *Comput. Mech.*, 52(6):1301–1312, 2013.
- [54] S. J. Cummins, M. M. Francois, and D. B. Kothe. Estimating curvature from volume fractions. *Computers & Structures*, 83(6):425–434, 2005.
- [55] B. Daly. A technique for including surface tension effects in hydrodynamics calculations. *J. Comput. Phys.*, 4:97–117, 1969.
- [56] J. Davies and E. Rideal. *Interfacial phenomena*. Academic Press, 1963.
- [57] S. de Groot and P. Mazur. *Non-Equilibrium Thermodynamics*. Dover, New York, 1st edition, 1984.

- [58] K. Deckelnick, C. Elliott, and V. Styles. Numerical diffusion-induced grain boundary motion. *Interfaces and Free Boundaries*, 3(4):393–414, 2001.
- [59] L. Dedè, M. Borden, and T. Hughes. Isogeometric analysis for topology optimization with a phase field model. *Archive Comp. Mech. Engrg.*, 19:427–465, 2012.
- [60] F. Denner and B. van Wachem. Fully-Coupled Balanced-Force VOF Framework for Arbitrary Meshes with Least-Squares Curvature Evaluation from Volume Fractions. *Numerical Heat Transfer, Part B: Fundamentals: An International Journal of Computation and Methodology*, 65(3):218–255, 2014.
- [61] J. Donea and A. Huerta. *Finite Element Methods for Flow Problems*. John Wiley & Sons, New York, 2003.
- [62] V. Doyeux, Y. Guyot, V. Chabannes, C. Prud’homme, and M. Ismaila. Simulation of two-fluid flows using a finite element/level set method. application to bubbles and vesicle dynamics. *Journal of Computational and Applied Mathematics*, 246:251–259, 2013.
- [63] G. Dziuk. An algorithm for evolutionary surfaces. *Numer. Math.*, 58(1):603–612, 1991.
- [64] G. Dziuk and C. Elliott. Finite elements on evolving surfaces. *SIAM*, 27(2):262–292, 2007.
- [65] G. Dziuk and C. Elliott. An Eulerian approach to transport and diffusion on evolving implicit surfaces. *Computing and Visualization in Science*, 13(1):17–28, 2010.
- [66] C. R. Easton. Homogeneous Boundary Conditions for Pressure in the MAC Method. *J. Comput. Phys.*, 9:375–379, 1972.
- [67] B. Echebarria, R. Folch, A. Karma, and M. Plapp. Quantitative phase-field model of alloy solidification. *Physical Review E*, 70:061604, 2004.
- [68] S. Elgeti, M. Probst, C. Windeck, M. Behr, W. Michaeli, and C. Hopmann. Numerical shape optimization as an approach to extrusion die design. *Finite Elements in Analysis and Design*, 48:35–43, 2012.
- [69] S. Elgeti, H. Sauerland, L. Pauli, and M. Behr. On the usage of NURBS as interface representation in free-surface flows. *Int. J. Numer. Methods Fluids*, 69(1):73–87, 2012.
- [70] H. Emmerich. *The Diffuse Interface Approach in Materials Science: Thermodynamic Concepts and Applications of Phase-Field Models*. Springer, 2003.
- [71] S. Engblom, M. Do-Quang, G. Amberg, and A.-K. Tornberg. On diffuse interface modeling and simulation of surfactants in two-phase fluid flow. *Communications in Computational Physics*, 14(4):879–915, 2013.
- [72] M. Engelman, R. Sani, and P. Gresho. The implementation of normal and/or tangential boundary conditions in finite element codes for incompressible fluid flow. *Int. J. Numer. Methods Fluids*, 2:225–238, 1982.

- [73] J.-H. Enschenburg and J. Jost. *Differentialgeometrie und Minimalflächen*. Springer, 2007.
- [74] A. Ern and J.-L. Guermond. Applied Mathematical Sciences. In S. Antman, J. Marsden, and L. Sirovich, editors, *Theory and Practice of Finite Elements*, volume 159. Springer, 2003.
- [75] J. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics*. Springer, 1999.
- [76] P. Fife, J. Cahn, and C. Elliott. A free-boundary model for diffusion-induced grain boundary motion. *Interfaces and Free Boundaries*, 3:291–336, 2001.
- [77] C. Frederiksen and A. Watts. Finite-element method for time-dependent incompressible free surface flows. *J. Comput. Phys.*, 39:282–304, 1981.
- [78] T. Fries. The intrinsic XFEM for two-fluid flows. *Int. J. Numer. Methods Fluids*, 60:437 – 471, 2008.
- [79] T. Fries and T. Belytschko. The extended/generalized finite element method: An overview of the method and its applications. *Internat. J. Numer. Methods Engrg.*, 84(3):253–304, 2010.
- [80] T. Fries, A. Byfut, A. Alizada, K. Cheng, and A. Schröder. Hanging nodes and xfem. *Internat. J. Numer. Methods Engrg.*, 86(4-5):404–430, 2011.
- [81] M. Fuchs, B. Jüttler, O. Scherzer, and H. Yang. Combined evolution of level sets and b-spline curves for imaging. Technical report, FSP Report No. 41, 2007.
- [82] S. Ganesan and L. Tobiska. Computations of Flows with Interfaces Using Arbitrary Lagrangian Eulerian Method. In P. Wesseling, E. Oñate, and J. Périaux, editors, *European Conference on Computational Fluid Dynamics*. TU Delft, The Netherlands, 2006.
- [83] S. Ganesan and L. Tobiska. A coupled arbitrary Lagrangian–Eulerian and Lagrangian method for computation of free surface flows with insoluble surfactants. *J. Comput. Phys.*, 228(8):2859–2873, 2009.
- [84] S. Ganesan and L. Tobiska. Modelling and simulation of moving contact line problems with wetting effects. *Computing and Visualization in Science*, 12(7):329–336, 2009.
- [85] S. Ganesan and L. Tobiska. Arbitrary Lagrangian–Eulerian finite-element method for computation of two-phase flows with soluble surfactants. *J. Comput. Phys.*, 231:3685–3702, 2012.
- [86] S. Ganesan, G. Matthies, and L. Tobiska. On spurious velocities in incompressible flow problems with interfaces. *Comp. Methods Appl. Mech. Engrg.*, 196(7):1193–1202, 2007.
- [87] S. Ganesan, G. Matthies, and L. Tobiska. On spurious velocities in incompressible flow problems with interfaces. *Comp. Methods Appl. Mech. Engrg.*, 196:1193 – 1202, 2007. ISSN 0045-7825.

- [88] I. Gavriluk, M. Hermann, I. Lukovsky, O. Solodun, and A. Timokha. Weakly nonlinear sloshing in a truncated circular conical tank. *Fluid Dynamic Research*, 45, 2013.
- [89] J. Gibbs. *On the Equilibrium of Heterogeneous Substances*. Transactions of the Connecticut Academy of Arts and Sciences, 1874-1878.
- [90] V. Girault. A Combined Finite Element and Marker and Cell Method for Solving Navier-Stokes Equations. *Numer. Math.*, 26:39–59, 1976.
- [91] H. Gómez, V. Calo, Y. Bazilevs, and T. Hughes. Isogeometric analysis of the Cahn–Hilliard phase-field model. *Comp. Methods Appl. Mech. Engrg.*, 197(49):4333–4352, 2008.
- [92] J. Gómez-Góñi, C. A. Garrido-Mendoza, J. L. Cercós, and L. González. Two phase analysis of sloshing in a rectangular container with Volume of Fluid (VOF) methods. *Ocean Engineering*, 73:208–212, 2013.
- [93] B. Gonzalez-Ferreiro, H. Gomez, and I. Romero. A thermodynamically consistent numerical method for a phase field model of solidification. *Communications in Nonlinear Science and Numerical Simulation*, 19(7):2309–2323, 2014.
- [94] A. Gray. *Modern differential geometry of curves and surfaces with Mathematica*. CRC Press, Boca Raton, 2nd edition, 1998. ISBN 0849371643 (alk. paper).
- [95] S. Groß and A. Reusken. An extended pressure finite element space for two-phase incompressible flows with surface tension. *J. Comput. Phys.*, 224(1):40 – 58, 2007.
- [96] S. Gross and A. Reusken. Finite element discretization error analysis of a surface tension force in two-phase incompressible flows. *SIAM*, 45(4):1679–1700, 2007.
- [97] S. Groß and A. Reusken. *Numerical Methods for Two-Phase Incompressible Flows*. Springer, 2010.
- [98] G. Grün and F. Klingbeil. Two-phase flow with mass density contrast: Stable schemes for a thermodynamic consistent and frame-indifferent diffuse-interface model. *J. Comput. Phys.*, 257(A):708–725, 2014.
- [99] I. Gyrmati. *Non-Equilibrium Thermodynamics*. Springer, New York, 1st edition, 1970.
- [100] P. Hansbo. The characteristic streamline diffusion method for the time-dependent incompressible Navier-Stokes equations. *Comp. Methods Appl. Mech. Engrg.*, 99:171–186, 1992.
- [101] P. Hansbo and A. Szepessy. A velocity-pressure streamline diffusion finite element method for the incompressible Navier-Stokes equations. *Comp. Methods Appl. Mech. Engrg.*, 84: 175–192, 1990.
- [102] F. Harlow, J. Welch, J. Shannon, and B. Daly. The MAC method. Report LA-3425, Los Alamos Scientific Laboratory, 1965.

- [103] D. Hartmann, M. Meinke, and W. Schröder. Differential equation based constrained reinitialization for level set methods. *J. Comput. Phys.*, 227(14):6821–6845, 2008.
- [104] M. Hayashi, K. Hatanaka, and M. Kawahara. Lagrangian finite element method for free surface Navier-Stokes flow using fractional step methods. *Int. J. Numer. Methods Fluids*, 13(7):805–840, 1991.
- [105] C. Hirt and B. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.*, 39:201 – 225, 1981.
- [106] C. Hirt, J. Cook, and T. Butler. A Lagrangian method for calculating the dynamics of an incompressible fluid with free surface. *J. Comput. Phys.*, 5:103–124, 1970. MAC versus Lagrangian; Lagrangian surface tension instabilities.
- [107] C. Hirt, A. Amsden, and J. Cook. An arbitrary Lagrangian Eulerian computing method for all flow speeds. *J. Comput. Phys.*, 14:227 – 253, 1974.
- [108] C. W. Hirt, B. D. Nichols, and N. C. Romero. SOLA: A numerical solution algorithm for transient fluid flows. Technical Report 32418, NASA STI/Recon Technical Report N 75, 1975.
- [109] B. Hübner, E. Walhorn, and D. Dinkler. A monolithic approach to fluid-structure interaction using space-time finite elements. *Comp. Methods Appl. Mech. Engrg.*, 193:2087–2104, 2004.
- [110] A. Huerta and W. Liu. Viscous flow with large free surface motion. *Comp. Methods Appl. Mech. Engrg.*, 69:277 – 324, 1988.
- [111] T. Hughes and G. Hulbert. Space-time finite element methods for elastodynamics: formulations and error estimates. *Comp. Methods Appl. Mech. Engrg.*, 66:339 – 363, 1988.
- [112] T. Hughes, W. Liu, and T. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Comp. Methods Appl. Mech. Engrg.*, 29:329–349, 1981.
- [113] T. J. R. Hughes. *The Finite Element Method*. Dover Publications Inc., 2000.
- [114] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comp. Methods Appl. Mech. Engrg.*, 194:4135–4195, 2005.
- [115] S. Hysing and S. Turek. The eikonal equation: numerical efficiency vs. algorithmic complexity on quadrilateral grids. In *Proceedings of Algoritmy 2005, Conference on Scientific Computing*, pages 22–31. Slovak University of Technology, Bratislava, ISBN 80-227-2192-1, 2005.
- [116] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, and L. Tobiska. Quantitative benchmark computations of two-dimensional bubble dynamics. *Int. J. Numer. Methods Fluids*, 60(11):1259–1288, 2009.

- [117] S. Idelsohn, M. Mier-Torrecilla, N. Nigro, and E. Oñate. On the analysis of heterogeneous fluids with jumps in the viscosity using a discontinuous pressure field. *Comput. Mech.*, 46(1):115–124, 2010.
- [118] K. Ito, T. Kunugi, H. Ohshima, and T. Kawamura. A volume-conservative PLIC algorithm on three-dimensional fully unstructured meshes. *Computers & Fluids*, 88:250–261, 2013.
- [119] A. James and J. Lowengrub. A surfactant-conserving volume-of-fluid method for interfacial flows with insoluble surfactant. *J. Comput. Phys.*, 201(2):685–722, 2004.
- [120] P. Jamet. Galerkin-type approximations which are discontinuous in time for parabolic equations in a variable domain. *SIAM*, 15(5):912–928, 1978.
- [121] P. Jamet and R. Bonnerot. Numerical solution of the Eulerian equations of compressible flow by a finite element method which follows the free boundary and the interfaces. *J. Comput. Phys.*, 18:21–45, 1975.
- [122] A. Johnson and T. Tezduyar. Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. *Comp. Methods Appl. Mech. Engrg.*, 119:73 – 94, 1994.
- [123] K. Kakuda, T. Nagashima, Y. Hayashi, S. Obara, J. Toyotani, S. Miura, N. Katsurada, S. Higuchi, and S. Matsuda. Three-dimensional fluid flow simulations using gpu-based particle method. *Comp. Mod. Engrg. Sci.*, 93(5):363–376, 2013.
- [124] M. Kang, R. Fedkiw, and X.-D. Liu. A boundary condition capturing method for multiphase incompressible flow. *Journal of Scientific Computing*, 15(3):323–360, 2000.
- [125] J. Kistler. *Hydrodynamics of Wetting*, volume Wettability. Dekker, 1993.
- [126] P. Kjellgren and J. Hyvarinen. An Arbitrary Lagrangian-Eulerian finite element method. *Comput. Mech.*, 21(1):81–90, 1998.
- [127] K. Kleefsman, G. Fekken, A. Veldman, B. Iwanowski, and B. Buchner. A volume-of-fluid based simulation method for wave impact problems. *J. Comput. Phys.*, 206(1):363–393, 2005.
- [128] S. Klinkel. Fortgeschrittene strukturanalysen. Technical report, RWTH Aachen University, 2012.
- [129] J. Klostermann, K. Schaake, and R. Schwarze. Numerical simulation of a single rising bubble by VOF with surface compression. *Int. J. Numer. Methods Fluids*, 71:960–982, 2013.
- [130] P. Knupp, L. Margolin, and M. Shashkov. Reference jacobian optimization-based rezone strategies for arbitrary lagrangian eulerian methods. *J. Comput. Phys.*, 176(1):93–128, 2002.

- [131] A. Kölke. *Modellierung und Diskretisierung bewegter Diskontinuitäten in randgekoppelten Mehrfeldsystemen*. PhD thesis, Technical University Braunschweig, 2005.
- [132] S. Koshizuka and Y. Oka. Moving-particle semi-implicit method for fragmentation of incompressible fluid. *Nuclear Science and Engineering*, 123:421–434, 1996.
- [133] Y.-G. Lee, K.-L. Jeong, and N. Kim. The marker-density method in cartesian grids applied to nonlinear ship waves. *Computers & Fluids*, 63:57–69, 2012.
- [134] C. Leung and M. Berzins. A computational model for organism growth based on surface mesh generation. *J. Comput. Phys.*, 188(1):75–99, 2003.
- [135] S. Leung and H. Zhao. A grid based particle method for moving interface problems. *J. Comput. Phys.*, 228(8):2993–3024, 2009.
- [136] S. Leung, J. Lowengrub, and H. Zhao. A grid based particle method for solving partial differential equations on evolving surfaces and modeling high order geometrical motion. *J. Comput. Phys.*, 230(7):2540–2561, 2011.
- [137] Z. Li and M.-C. Lai. The immersed interface method for the Navier-Stokes equations with singular forces. *J. Comput. Phys.*, 171(2):822–842, 2001.
- [138] Z. Li, F. Jaber, and T. Shih. A hybrid Lagrangian-Eulerian particle-level set method for numerical simulations of two-fluid turbulent flows. *Int. J. Numer. Methods Fluids*, 56: 2271–2300, 2008.
- [139] P. Liovic, M. Francois, M. Rudman, and R. Manasseh. Efficient simulation of surface tension-dominated flows through enhanced interface geometry interrogation. *J. Comput. Phys.*, 229:7520–7544, 2010.
- [140] H. Liu and Y. Zhang. Phase-field modeling droplet dynamics with soluble surfactants. *J. Comput. Phys.*, 229(24):9166–9187, 2010.
- [141] E. Lopez, N. Nigro, and M. Storti. Simultaneous untangling and smoothing of moving grids. *Internat. J. Numer. Methods Engrg.*, 76(7):994–1019, 2008.
- [142] J. López, J. Hernández, P. Gómez, and F. Faura. A volume of fluid method based on multidimensional advection and spline interface reconstruction. *J. Comput. Phys.*, 195: 718–742, 2004.
- [143] J. López, C. Zanzi, P. Gómez, R. Zamora, F. Faura, and J. Hernández. An improved height function technique for computing interface curvature from volume fractions. *Comp. Methods Appl. Mech. Engrg.*, 198(33):2555–2564, 2009.
- [144] R. Loubère, P. Maire, M. Shashkov, J. Breil, and S. Galera. Reale: A reconnection-based arbitrary-lagrangian-eulerian method. *J. Comput. Phys.*, 229(12):4724–4761, 2010.
- [145] J. Lowengrub, J.-J. Xu, and A. Voigt. Surface phase separation and flow in a simple model of multicomponent drops and vesicles. *Fluid Dynamics and Materials Processing*, 3(1): 1–20, 2007.

- [146] K. Lucas. *Thermodynamik*. Springer, Berlin Heidelberg, 2007.
- [147] R. Luppens, A. Veldman, and R. Wemmenhove. Simulation of two-phase flow in sloshing tanks. In *Computational Fluid Dynamics*, pages 555–561. Springer, 2011.
- [148] D. Lynch and W. Gray. Finite element simulation of flow in deforming regions. *J. Comput. Phys.*, 36:135–153, 1980.
- [149] E. Marchandise, P. Geuzaine, N. Chevaugeon, and J.-F. Remacle. A stabilized finite element method using a discontinuous level set approach for the computation of bubble dynamics. *J. Comput. Phys.*, 225(1):949–974, 2007.
- [150] M. Marek, W. Aniszewski, and A. Boguslawski. Simplified volume of fluid method (SVOF) for two-phase flows. *TASK Quarterly*, 12(3):255–265, 2008.
- [151] H. Marschall, S. Boden, C. Lehrenfeld, U. Hampel, A. Reusken, M. Wörner, and D. Bothe. Validation of interface capturing and tracking techniques with different surface tension treatments against a taylor bubble benchmark problem. *Computers and Fluids*, 102:336–352, 2014.
- [152] A. Masud and T. Hughes. A space-time Galerkin/least-squares finite element formulation of the Navier-Stokes equations for moving domain problems. *Comp. Methods Appl. Mech. Engrg.*, 146:91–126, 1997.
- [153] T. Matsui and T. Nagaya. Nonlinear sloshing in a floating-roofed oil storage tank under long-period seismic ground motion. *Earthquake Engineering and Structural Dynamics*, 42:973–991, 2013.
- [154] S. McKee, M. Tomé, J. Cuminato, A. Castelo, and V. Ferreira. Recent advances in the marker and cell method. *Archive Comp. Mech. Engrg.*, 11(2):107–142, 2004.
- [155] M. Meier, G. Yadigaroglu, and B. Smith. A novel technique for including surface tension in PLIC-VOF methods. *European Journal of Mechanics B: Fluids*, 21:61–73, 2002.
- [156] J. Mencinger and I. Zun. A PLIC-VOF method suited for adaptive moving grids. *J. Comput. Phys.*, 230:644–663, 2011.
- [157] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. P. The Google Books Team, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, M. A. Nowak, and E. L. Aiden. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.
- [158] M. Mier-Torrecilla, S. Idelsohn, and E. Oñate. Advances in the simulation of multi-fluid flows with the particle finite element method. application to bubble dynamics. *Int. J. Numer. Methods Fluids*, 67:1516–1539, 2011.
- [159] P. Mineev, T. Chen, and K. Nandakumar. A finite element technique for multifluid incompressible flow using eulerian grids. *J. Comput. Phys.*, 187(1):255–273, 2003.

- [160] S. Mittal and T. Tezduyar. A finite element study of incompressible flows past oscillating cylinders and airfoils. *Int. J. Numer. Methods Fluids*, 15:1073–1118, 1992.
- [161] N. Moës, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *Internat. J. Numer. Methods Engrg.*, 46(1):131–150, 1999.
- [162] G. Mompean, L. Thais, M. Tomé, and A. Castelo. Numerical prediction of three-dimensional time-dependent viscoelastic extrudate swell using differential and algebraic models. *Computers & Fluids*, 44:68–78, 2011.
- [163] F. Mut, G. Buscaglia, and E. Dari. New mass-conserving algorithm for level set redistancing on unstructured meshes. *J. Appl. Mech., ASME*, 73(6):1011–1016, 2006.
- [164] B. Nichols and C. Hirt. Improved free surface boundary conditions for numerical incompressible-flow calculations. *J. Comput. Phys.*, 8(3):434–448, 1971.
- [165] W. Noh and P. Woodward. SLIC (Simple Line Interface Calculation). *Lecture Notes in Physics*, 59:330–340, 1976.
- [166] E. Oñate and J. García. A finite element method for fluid-structure interaction with surface waves using a finite calculus formulation. *Comp. Methods Appl. Mech. Engrg.*, 191:635–660, 2001.
- [167] E. Oñate, S. Idelsohn, F. D. Pin, and R. Aubry. The particle finite element method: an overview. *International Journal of Computational Methods*, 1(2):267–307, 2004.
- [168] T. Okamoto and M. Kawahara. Two-dimensional sloshing analysis by Lagrangian finite element method. *Int. J. Numer. Methods Fluids*, 11:453–477, 1990.
- [169] M. Olshanskii, A. Reusken, and J. Grande. A finite element method for elliptic equations on surfaces. *SIAM*, 47(5):3339–3358, 2009.
- [170] E. Olsson, G. Kreiss, and S. Zahedi. A conservative level set method for two phase flow ii. *J. Comput. Phys.*, 225:785–807, 2007.
- [171] L. Onsager. Reciprocal Relations in Irreversible Processes. *Physical Review*, 37(405), 1931.
- [172] S. Osher and R. Fedkiw. *Level set methods and dynamic implicit surfaces*, volume Applied mathematical sciences. Springer, New York, 2003.
- [173] S. Osher and J. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton Jacobi formulations. *J. Comput. Phys.*, 79:12–49, 1988.
- [174] Z. Ozdemir, M. Souli, and Y. Fahjan. FSI methods for seismic analysis of sloshing tank problems. *Mechanics and Industry*, 11:133–147, 2010.
- [175] F. Pasenow, A. Zilian, and D. Dinkler. XFEM coupling of granular flows interacting with surrounding fluids. In *ECCOMAS 2012-European Congress on Computational Methods in Applied Sciences and Engineering e-Book Full Papers*, 2012.

- [176] L. Pauli, M. Behr, and S. Elgeti. Towards shape optimization of extrusion dies with respect to homogeneous die swell. *Journal of Non-Newtonian Fluid Mechanics*, 69:73–87, 2012.
- [177] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang. A pde-based fast local level set method. *J. Comput. Phys.*, 155(2):410–438, 1999.
- [178] L. Piegel and W. Tiller. *The NURBS Book*. Springer, Berlin, Germany, 1997.
- [179] J. Pilliod and E. Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J. Comput. Phys.*, 199:465–502, 2004.
- [180] T. Plewa, T. Linde, and V. Weirs. Adaptive mesh refinement - theory and applications. In *Proceedings of the Chicago Workshop on Adaptive Mesh Refinement Methods, Sept. 3-5, 2003*. Lecture Notes in Computational Science and Engineering, Springer, 2005.
- [181] I. Prigogine. *Non-Equilibrium Statistical Mechanics*. Wiley, New York, 2nd edition, 1966.
- [182] M. Probst. *Robust Shape Optimization for Incompressible Flow of Shear-Thinning Fluids*. PhD thesis, RWTH Aachen University, 2013.
- [183] R. Radovitzky and M. Ortiz. Lagrangian finite element analysis of newtonian fluid flows. *Internat. J. Numer. Methods Engrg.*, 43(4):607–619, 1998.
- [184] M. Raessi, J. Mostaghimi, and M. Bussmann. A volume-of-fluid interfacial flow solver with advected normals. *Computers and Fluids*, 39(8):1401–1410, 2010.
- [185] U. Rasthofer, F. Henke, W. Wall, and V. Gravemeier. An extended residual-based variational multiscale method for two-phase flow including surface tension. *Comp. Methods Appl. Mech. Engrg.*, 200(21–22):1866–1876, 2011.
- [186] A. Reusken. Analysis of an extended pressure finite element space for two-phase incompressible flows. *Computing and Visualization in Science*, 11:293–305, 2008.
- [187] A. Reusken and E. Loch. On the accuracy of the level set supg method for approximating interfaces. Technical report, Institut für Geometrie und Praktische Mathematik, 2011.
- [188] W. Rider and D. Kothe. Reconstructing volume tracking. *J. Comput. Phys.*, 141:112–152, 1998.
- [189] D. Rogers. *An Introduction to NURBS with Historical Perspective*. Morgan Kaufmann Publishers, 2001.
- [190] F. Santos, V. Ferreira, M. Tomé, A. Castelo, N. Mangiavacchi, and S. McKee. A marker-and-cell approach to free surface 2-d multiphase flows. *Int. J. Numer. Methods Fluids*, 70:1543–1557, 2012.
- [191] H. Sauerland. *An XFEM Based Sharp Interface Approach for Two-Phase and Free-Surface Flows*. PhD thesis, RWTH Aachen University, 2013.

- [192] H. Sauerland and T. Fries. The extended finite element method for two-phase and free-surface flows. *J. Comput. Phys.*, 230(9):3369–3390, 2011.
- [193] C. Schroeder, W. Zheng, and R. Fedkiw. Semi-implicit surface tension formulation with a lagrangian surface mesh on an eulerian simulation grid. *J. Comput. Phys.*, 231:2092–2115, 2012.
- [194] R. Sevilla, S. Fernandez-Mendez, and A. Huerta. NURBS-Enhanced Finite Element Method (NEFEM). *Internat. J. Numer. Methods Engrg.*, 76(1):56–83, 2008.
- [195] R. Sevilla, S. Fernandez-Mendez, and A. Huerta. NURBS-Enhanced Finite Element Method (NEFEM): A Seamless Bridge Between CAD and FEM. *Archive Comp. Mech. Engrg.*, 18(4):441–484, 2011.
- [196] R. Sevilla, S. Fernandez-Mendez, and A. Huerta. 3D NURBS-Enhanced Finite Element Method (NEFEM). *Internat. J. Numer. Methods Engrg.*, 88:103–125, 2011.
- [197] W. Silliman and L. Scriven. Slip of liquid inside a channel exit. *Physics of Fluids*, 21(11): 2115–2116, 1978.
- [198] I. Singer-Loginova and H. Singer. The phase field technique for modeling multiphase materials. *Reports on Progress in Physics*, 71, 2008.
- [199] R. Siquieri and H. Emmerich. Phase-field investigation of microstructure evolution under the influence of convection. *Philosophical Magazine*, 91(1):45–73, 2011.
- [200] A. A. Sliwiak. Free-surface description in a sloshing tank with deformable shape. Bachelor’s thesis, 2014.
- [201] F. Sousa, R. Ausas, and G. Buscaglia. Numerical assessment of stability of interface discontinuous finite element pressure spaces. *Comp. Methods Appl. Mech. Engrg.*, 245: 63–74, 2012.
- [202] A. Stavrev, P. Knechtges, S. Elgeti, and A. Huerta. Space-time nurbs-enhanced finite elements for free-surface flows in 2d. *to be submitted to Int. J. Numer. Methods Fluids*, 2015.
- [203] I. Steinbach. Phase-field models in materials science. *Modelling and Simulation in Material Science and Engineering*, 17(7), 2009.
- [204] M. Sussman. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comput. Phys.*, 187:110–136, 2003.
- [205] M. Sussman and E. Puckett. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. *J. Comput. Phys.*, 162:301–337, 2000.
- [206] M. Sussman, A. Almgren, J. Bell, P. C. L. H. Howell, and M. L. Welcomey. An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys.*, 148:81–124, 1999.

- [207] T. E. Tezduyar, M. Behr, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: I. the concept and the preliminary numerical tests. *Comp. Methods Appl. Mech. Engrg.*, 94(3):339 – 351, 1992.
- [208] T. E. Tezduyar, M. Behr, S. Mittal, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: II. computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. *Comp. Methods Appl. Mech. Engrg.*, 94(3):353 – 371, 1992.
- [209] M. Tomé and S. McKee. Gensmac: A computational marker and cell method for the free surface flows in general domains. *J. Comput. Phys.*, 110:171–186, 1994.
- [210] M. Tomé, A. Castelo, J. Cuminato, N. Mangiavacchi, and V. Ferreira. Gensmac3d: a numerical method for solving unsteady three-dimensional free surface flows. *Int. J. Numer. Methods Fluids*, 37:747–796, 2001.
- [211] M. Tomé, A. Castelo, V. Ferreira, and S. McKee. A finite difference technique for solving the Oldroyd-B model for 3D-unsteady free surface flows. *Journal of Non-Newtonian Fluid Mechanics*, 154:179–206, 2008.
- [212] M. Tomé, A. Castelo, A. Afonso, M. Alves, and F. Pinho. Application of the log-conformation tensor to three-dimensional time-dependent free surface flows. *Journal of Non-Newtonian Fluid Mechanics*, 2012.
- [213] D. Torres and J. Brackbill. The point-set method: front-tracking without connectivity. *J. Comput. Phys.*, 180:427–470, 2000.
- [214] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Taubner, J. Han, S. Nas, and Y. Jan. A front-tracking method for the computations of multi-phase flow. *J. Comput. Phys.*, 169(2):708–759, 2001.
- [215] A. Vaikuntam. *Numerical Estimation of Surface Parameters by Level Set Methods*. PhD thesis, University of Kaiserslautern, 2008.
- [216] W. Wall. *Fluid–Struktur–Interaktion mit stabilisierten Finiten Elementen*. PhD thesis, University of Stuttgart, 1999.
- [217] X. Wang and X. Li. Numerical simulation of three dimensional non-newtonian free surface flows in injection molding using ale finite element method. *Finite Elem. Anal. Des.*, 46(7): 551–562, 2010.
- [218] Z. Wang, J. Yang, and F. Stern. A new volume-of-fluid method with a constructed distance function on general structured grids. *J. Comput. Phys.*, 231:3703–3722, 2012.
- [219] M. Williams. *Numerical methods for tracking interfaces with surface tension in 3D mold filling processes*. PhD thesis, University of California, Davis, 2000.

- [220] F. Xiao, Y. Honma, and T. Kono. A simple algebraic interface capturing scheme using hyperbolic tangent function. *Int. J. Numer. Methods Fluids*, 48:1023–1040, 2005.
- [221] J.-J. Xu and H.-K. Zhao. An Eulerian formulation for solving partial differential equations along a moving interface. *Journal of Scientific Computing*, 19(1):573–594, 2003.
- [222] K. Yokoi. Efficient implementation of THINC scheme: a simple and practical smoothed VOF algorithm. *J. Comput. Phys.*, 226(2):1985–2002, 2007.
- [223] D. Youngs. An interface tracking method for a 3D Eulerian hydrodynamics code. Technical Report 44/92/35, AWRE, 1984.
- [224] W. Zheng, B. Zhu, B. Kim, and R. Fedkiw. A new incompressibility discretization for a hybrid mac grid representation with surface tension. *Int. J. Numer. Methods Fluids*, 62(9): 963–993, 2010.
- [225] S. Zlotnik and P. Díez. Hierarchical X-FEM for n-phase flow ($n > 2$). *Comp. Methods Appl. Mech. Engrg.*, 198(30–32):2329–2338, 2009.